Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Způsoby návrhu databáze nástrojem MySQL Workbench

Plzeň, 2015 Jana Nosková

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracovala samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 13. srpna 2015

Jana Nosková

Poděkování

Ráda bych poděkovala vedoucímu bakalářské práce Ing. Martinu Zímovi, Ph.D. za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování bakalářské práce.

Abstrakt

Název: Způsoby návrhu databáze nástrojem MySQL Workbench

Tato práce obsahuje manuál pro modelování dat nástrojem MySQL Workbench. Podrobněji jsou popsány různé způsoby tvorby fyzického datového modelu čili databáze. Všechny cesty byly otestovány vybranými subjekty. Díky tomuto testování bylo vytvořeno zhodnocení jednotlivých způsobů a nalezení toho, který je nejvhodnější pro skupiny uživatelů s různými znalostmi.

Klíčová slova: datový model, návrh databáze, integritní omezení

Abstract

Title: Methods of database design by tool MySQL Workbench

This document contains a manual for data modeling by database tool MySQL Workbench. The ways of database creation are described in detail. These methods were tested by chosen subjects. The final review was made based on this testing and the most suitable method was chosen for users with different knowledge.

Keywords: data model, database design, integrity constraints

Obsah

Seznam obrázků

1 Úvod

Tato bakalářská práce se zabývá způsoby tvorby databáze nástrojem MySQL Workbench. První stránky jsou proto věnovány základům systému MySQL, od vymezení použitých pojmů, přes definici vlastností jednotlivých objektů až po základní operace. Cílem je seznámit každého čtenáře, tedy i toho, který se s MySQL ještě nesetkal, alespoň okrajově s tímto databázovým systémem, na jehož základě je postavena celá bakalářská práce.

Další částí dokumentu je uživatelská příručka k nástroji MySQL Workbench. Je však omezena pouze na modelování dat, což je vlastní cíl práce. Cílem kapitoly věnující se MySQL Workbench je usnadnit čtenáři práci se studováním funkcionality popisovaného databázového nástroje.

Druhá část dokumentu popisuje jednotlivé způsoby, kterými lze vytvořit databázi. Jsou zde popsány celkem čtyři cesty. Protože jednotlivé postupy s sebou nesou různé překážky, obsahuje každý popsaný způsob seznam možných problémů a jejich řešení.

Jednotlivé způsoby tvorby databáze byly otestovány a zhodnoceny několika dobrovolníky různých znalostí. Úkolem dobrovolníků bylo vytvořit ukázkovou databázi všemi popsanými způsoby a výsledek popsat v dotazníku. Na základě otestování popsaných cest bylo vytvořeno zhodnocení a vybrán nejvhodnější způsob pro různé typy uživatelů na základě jejich znalostí.

V přílohách lze najít například grafický návrh modelované databáze, dotazník pro vyplnění výsledků testování či postup instalace a základní konfigurace nástroje MySQL Workbench.

2 Úvod do MySQL

MySQL je jeden z nejpoužívanějších databázových systémů. Je schopen ukládat velké množství různých dat a vracet je zpět v požadované podobě a čase. V současné době je vlastněný společností Oracle.

Tento systém je dostupný ve dvou verzích. Community Edition je tzv. open source[1], je zdarma a poskytuje základní funkcionalitu. Enterprise Edition je komerční verze s rozšířenou funkcionalitou, dostupná za poplatek. [3]

MySQL lze integrovat s jazyky PHP, Perl, Java, C, C++ a Python.[2] Může běžet na operačních systémech Windows, Linux i OS X. [5]

Jako všechny relační databázové systémy i MySQL využívá pro práci s databázemi strukturovaný dotazovací jazyk SQL (Structured Query Language).

2.1 Terminologie

Databáze značí modelovaný systém z reálného světa.

Druhy objektů sledovaného systému lze rozdělit do skupin čili *entit*. V databázi jsou entity reprezentovány *tabulkami*. Vztahy mezi entitami lze znázornit *relacemi*. Řádky tabulek – neboli *záznamy* – představují jednotlivé objekty. Ty jsou popsány svými vlastnostmi, které jsou zachyceny pomocí sloupců neboli *atributů*.
Každý záznam by měl být jedinečný. Jednotlivé atributy mohou být různých typů (více o datových typech viz kapitola 2.2).

Důležitým pojmem je *primární klíč*. Jedná se o atribut či množinu atributů, s jejichž pomocí se jednoznačně identifikují jednotlivé záznamy (řádky) v tabulce. Hodnoty primárních klíčů jsou jedinečné a žádná nesmí být prázdná. Obvykle se do tabulky zavádí jeden atribut navíc (většinou pojmenovaný ID), který plní pouze funkci primárního klíče. Pomocí tohoto atributu přiřadíme každému záznamu automaticky vygenerovaný klíč, většinou nezáporné celé číslo.

[1] Jedná se o vlastní zdrojový kód. Programovací kód tvořící MySQL je volně k dispozici veřejnosti – lidé mohou přispívat, opravovat chyby, vylepšovat program, navrhovat optimalizace.

Cizí klíč je atribut či množina atributů v tabulce, kterým odpovídá na základě relace primární klíč jiné tabulky.

Integrita databáze představuje stav, kdy jsou data platná, přesná a konzistentní. [2]

Databázové *schéma* označuje rezervovaný prostor na databázovém serveru (více o serveru viz Příloha B), ve kterém může uživatel vytvářet, modifikovat či mazat databázové objekty (především tabulky).

Při modelování databáze rozlišujeme dva typy datových modelů: fyzický a logický. *Logický datový model* označuje návrh databáze, kdy jednotlivé databázové objekty nejsou skutečně založeny. *Fyzický datový model* představuje databázi, jejíž objekty jsou již fyzicky založené na databázovém serveru.

Transakce je posloupnost souvisejících instrukcí, se kterými je nutné zacházet jako s jedním nedělitelným celkem. To znamená, že v transakci musí být provedeny buď všechny úlohy, nebo musí zůstat vše neprovedeno.

2.1.1 Relace

Relace značí vztahy mezi entitami (tabulkami). Dle kardinality vztahu rozlišujeme tři typy vazeb: 1:1, 1:N, M:N. U relace 1:1 či 1:N se navíc musí rozhodnout, zda bude tzv. *identifying* či *non-identifying*. Dle povinnosti vazby dále rozlišujeme relace povinné a nepovinné.

Relaci 1:1 používáme, pokud záznamu v tabulce A odpovídá maximálně jeden záznam v tabulce B a naopak. Tuto vazbu bychom mohli použít na vztah Prezident – Stát, tedy každý stát má v daném okamžiku pouze jednoho prezidenta. Zároveň však platí, že jeden prezident může v daném okamžiku vládnout pouze jednomu státu.

Relace 1:N znamená, že jednomu záznamu v tabulce A odpovídá více záznamů v tabulce B. Naopak jednomu záznamu v tabulce B je přiřazen maximálně jeden záznam z tabulky A. Například každá zdravotní pojišťovna má více klientů, ale každý člověk musí být zaregistrován pouze u jedné zdravotní pojišťovny.

Vztah M:N umožňuje každému záznamu v tabulce A přiřadit více záznamů z tabulky B a naopak. Tento vztah je nutné realizovat pomocnou, neboli rozkladovou, tabulkou.

Relace musí být rozložena do dvou vztahů 1:N. Relaci M:N lze aplikovat na vazbu Učitel – Student, kdy každý učitel učí více studentů a každý student má několik učitelů (na různé předměty).

Pro relaci, která je *identifying,* obecně platí, že primární klíč tabulky A je cizím klíčem tabulky B a zároveň součástí primárního klíče tabulky B. V nástroji MySQL Workbench je tento typ relace znázorněn plnou čarou.

Pro relaci *non-identifying* platí, že primární klíč tabulky A je v tabulce B jen cizím klíčem. V MySQL Workbench je tento typ relace znázorněn přerušovanou čarou.

2.2 Datové typy

Každý sloupec má přiřazený některý z datových typů. Rozlišujeme tři základní typy: číselné, řetězcové, datum a čas. Je vhodné volit takový typ sloupce, který poskytuje co nejmenší potřebný rozsah hodnot. Ovšem pozor na volbu typu sloupce s příliš malým rozsahem – může se stát, že data nepůjdou uložit.

2.2.1 Číselné typy

Jsou určeny pro ukládání jakýchkoli číselných dat. Může se jednat o celá čísla i čísla s desetinnou čárkou. Také lze zvolit, zda se číslo uloží bez znaménka (`UNSIGNED`) či se znaménkem (`SIGNED`). Jestliže je jakékoli pole `UNSIGNED`, znamená to, že neumožňuje uložit záporná čísla. Zároveň však rozšiřuje rozsah hodnot kladných celých čísel. [2]

INT či INTEGER

Jedná se o celé číslo umožňující rozsah hodnot od -2 147 483 648 do +2 147 483 647, bez znaménka 0 až 4 294 967 295. Tento datový typ má několik variací: `TINYINT` (rozsah 2^8 hodnot), `SMALLINT` (rozsah 2^{16} hodnot), `MEDIUMINT` (rozsah 2^{24} hodnot), `BIGINT` (rozsah 2^{64} hodnot).

FLOAT, DOUBLE

Tento číselný typ představuje číslo s plovoucí desetinnou čárkou. [5]

DEC (m, d) či DECIMAL (m, d)

Jedná se o číslo s pevnou desetinnou čárkou. Rozsah lze nastavit parametry *m* (počet číslic včetně des. čárky) a *d* (počet desetinných míst).

2.2.2 Řetězcové typy

Tento typ se používá pro sloupce, do kterých chceme ukládat libovolná znaková data, například jména, adresy atd. Velikost zabrané paměti se odvíjí od velikosti zadaného řetězce.

CHAR

Tento datový typ slouží k ukládání řetězců s pevnou délkou. Za typem `CHAR` obvykle následuje délka řetězce, například `CHAR(10)`. Není-li definována délka, automaticky se nastaví `CHAR(1)`. Maximální délka tohoto datového typu je 255 znaků. Hodnoty typu `CHAR` budou mít při ukládání vždy přesně takovou délku, jaká je specifikována, protože obsah sloupce se doplní mezerami. Ty se při vyplňování sloupce automaticky odstraní.

VARCHAR

Do tohoto datového typu se ukládají řetězce s proměnlivou délkou. Je-li vložený řetězec kratší než nastavíme, chybějící znaky se nedoplňují. Za typem je nutné specifikovat délku, například `VARCHAR(20)`.

TEXT

Jedná se o datový typ pro ukládání delších částí textu, které se nevejdou do typů `CHAR` či `VARCHAR`. [5]

2.2.3 Typ datum a čas

V případě nutnosti ukládat časové údaje použijeme sloupce typu datum a čas. Jedná se například o datum narození, konkrétní čas atd.

DATE, TIME

Datový typ `DATE` slouží k ukládání kalendářního data. Zobrazeno je ve tvaru YYYY-MM-DD. `TIME` slouží k uložení času, který je zobrazen ve tvaru HH:MM:SS.

DATETIME

Tento datový typ je kombinací předchozích dvou typů. Ukládá datum i čas a zobrazuje je ve tvaru YYYY-MM-DD HH:MM:SS. [5]

2.2.4 Ostatní datové typy

BLOB

`BLOB` je podobný datovému typu `TEXT`. Má stejný rozsah i velikost, je však určený k ukládání binárních dat, například souborů, obrázků.

2.3 Integritní omezení

Pojmem integritní omezení jsou označena pravidla, která pomáhají definovat integritu databáze – tedy dovolí uživateli zadat pouze taková data, která vyhovují předem definovaným kritériím. Existují tři druhy integritních omezení: entitní, doménové a referenční.

Entitní integritní omezení se používá pro specifikaci primárního klíče tabulky. Pomáhá také při označení atributů, které musí obsahovat nějakou hodnotu, nebo při definici jedinečných hodnot.

Doménová integrita umožňuje definovat omezení na úrovni hodnot sloupců. Představme si například omezení rozsahu hodnot.

Referenční integritní omezení je mezitabulkovou záležitostí. Definuje vztah mezi dvěma tabulkami s jedním cizím klíčem.[4]

Většina databázových systémů sama nutí uživatele dodržovat entitní a doménovou integritu. Na referenční integritu však musíme dbát sami.

V nástroji MySQL Workbench lze nastavit tato integritní omezení:

- `NN (NOT NULL)` – sloupec s touto vlastností bude muset v každé buňce obsahovat nějakou hodnotu.

- `PK (PRIMARY KEY)` – takto označený sloupec je součástí primárního klíče tabulky. Pokud v nástroji MySQL Workbench definujeme některému z atributů tuto vlastnost, automaticky je mu přiřazena i vlastnost `NOT NULL`, protože primární klíč musí vždy obsahovat nějakou hodnotu.
- `UQ (UNIQUE)` – v daném sloupci nesmějí být v buňkách stejné hodnoty. Každý řádek bude obsahovat v tomto atributu unikátní hodnotu.
- `BIN (BINARY)` – takto označený atribut bude obsahovat pouze hodnoty v binární podobě (binární data).
- `UN (UNSIGNED)` – povoluje pouze nezáporné číselné hodnoty.
- `ZF (ZERO FILL)` – hodnoty ve sloupci s touto vlastností budou zobrazovány s určitým počtem nul na začátku tak, aby byla vždy zobrazena celá definovaná šířka (například u datového typu `INT(4)` by bylo číslo 1 zobrazeno jako 0001)

2.4 Jiné vlastnosti

Databázový systém MySQL nabízí také možnost nastavení vlastnosti `AI (AUTO_INCREMENT)`. Znamená to, že systém si sám ve sloupci generuje jedinečné číselné hodnoty. Tuto vlastnost lze použít pouze na celočíselný datový typ. Navíc lze určit výchozí hodnotu.

2.5 Základní příkazy

Jak již bylo zmíněno, systém MySQL využívá pro definici dat a manipulaci s nimi jazyk SQL. Ten je rozdělen do čtyř různých jazyků podle funkcionality.

Jazyk DDL (Data Definition Language) slouží pro definici dat. Patří sem příkazy pro vytvoření databáze, tabulky či pohledu, změnu struktury a vymazání tabulky či pohledu. Každý příkaz tohoto jazyka zároveň funguje jako potvrzení všech předešlých transakcí.

Dále existuje jazyk DML (Data Manipulation Language), který je používán pro manipulaci s daty. Například pro vkládání či mazání záznamů, úpravu vložených záznamů či pro tvorbu dotazů.

DCL (Data Control Language) obsahuje příkazy pro udělení či odebrání práv.

Jazyk TCL (Transaction Control Language) se používá pro řízení transakcí.

Co se týká samotných příkazů, MySQL není v tomto ohledu tzv. case-sensitive. To znamená, že nezáleží na tom, zda píšeme příkaz velkými či malými písmeny. Například příkaz na vytvoření tabulky `CREATE TABLE` lze napsat jako `create table` či `Create Table` apod. Názvy tabulek i atributů píšeme raději bez diakritiky. Mohl by být problém s nastavením správného kódování a názvy by se nezobrazovaly správně. Názvy objektů databáze jsou na rozdíl od příkazů case-sensitive.

Příkazy můžeme rozložit do více řádků. Na konci každého příkazu musí být středník. [1]

Existuje několik základních příkazů. Obecně jsou to:

DDL

`CREATE`	–	vytváří databázové objekty (tabulky, pohledy,…)
`DROP`	–	odstraňuje databázové objekty
`ALTER`	–	změní definici atributů v tabulce, umožňuje přejmenovat tabulku, změnit integritní omezení či jiné vlastnosti (například AUTO_INCREMENT) a přidat či odebrat atributy
	–	tento příkaz je dobré používat jen tehdy, je-li změna prováděna na prázdné tabulce, tedy na takové, která ještě není naplněna daty

DML

`INSERT`	–	vkládá záznamy do tabulky
`UPDATE`	–	mění data v tabulce
`SELECT`	–	používá se při tvorbě dotazů – vrací výsledky dotazu
`DELETE`	–	smaže/odstraňuje záznamy z tabulky

DCL

`GRANT` – používá se pro přidělení práv

`REVOKE` – odebrání práv

TCL

`COMMIT` – potvrzení transakce

`ROLLBACK` – zrušení transakce

2.6 Typy tabulek – enginy

Protože velmi záleží na vnitřní reprezentaci dat v databázi, vzniklo několik typů tabulek. Každý z nich má své výhody i nevýhody. U některých důležitých změn v databázi musíme zajistit, aby se množina dotazů vykonala buď celá, nebo se nestalo nic. Nesmí se v průběhu vykonávání příkazů stát, že dojde k chybě a nebude vše dokončeno. Řešení nabízí transakce, což je vlastně posloupnost instrukcí, se kterými je nutné zacházet jako s nedělitelným celkem. Transakce vždy zahájíme, vykonáme příkazy a celou transakci potvrdíme příkazem `COMMIT` či zrušíme příkazem `ROLLBACK`. Ostatní uživatelé neuvidí změnu až do potvrzení transakce.

Existují dva transakčně bezpečné typy tabulek (InnoDB, BDB), ostatní už transakčně bezpečné nejsou (ISAM, MyISAM, MERGE, HEAP). Nejčastěji se používají enginy InnoDB a MyISAM. Hlavní odlišnost těchto dvou enginů se týká cizích klíčů. MyISAM sice dovolí definovat atribut, jenž by měl sloužit jako cizí klíč, ale nerespektuje ho. Po naplnění tabulky nebudeme mít kontrolu, že daný záznam existuje. Chceme-li zajistit referenční integritu, použijeme typ tabulek InnoDB.

3 MySQL Workbench

Tato kapitola obsahuje osobní poznatky a postřehy, bylo však čerpáno také z manuálu k nástroji MySQL Workbench [4].

MySQL Workbench je jednotný vizuální nástroj pro databázové architekty, vývojáře a databázové administrátory. Umožňuje přístup k datům umístěným na MySQL serveru[2]. S jeho pomocí lze navrhovat, vytvářet i konfigurovat databáze, ale také například konfigurovat a spravovat server, provádět administraci uživatelů či zálohovat databázi/data. [3]

MySQL Workbench podporuje MySQL od verze 5.1 a výš. Funguje na operačních systémech Windows, Linux a Mac OS X.

Tento databázový nástroj je dostupný ve dvou verzích. Community Edition je zdarma a obsahuje základní funkcionalitu, zatímco za Enterprise Edition, která poskytuje některé funkce navíc, je nutné zaplatit. V současné době vyšla verze 6.3. Při tvorbě této práce byla využita verze 6.0 CE.

Funkcionalitu tohoto nástroje lze rozdělit do pěti hlavních skupin:

1) **SQL Development:** umožňuje vytvářet či mazat fyzické databáze, modifikovat je a vkládat do nich data. S tímto souvisí možnost provádět nad databází dotazy.
2) **Data Modeling (Design):** poskytuje zejména funkce pro tvorbu grafických modelů databázových schémat, tedy návrhů databází.
3) **Server Administration**: umožňuje spravovat MySQL server a vést administraci uživatelů. Mezi další funkcionalitu lze zařadit také pravidelné zálohování, tvorbu logů či monitorování chování serveru. Tato funkcionalita není v dokumentu popsána.
4) **Data Migration:** poskytuje funkcionalitu pro migraci dat z jiných databázových systémů, jako jsou Microsoft SQL Server, Microsoft Access, Sybase ASE, SQLite, SQL Anywhere, PostgreSQL. Lze migrovat také data z dřívějších verzí MySQL.

[2] Více o MySQL serveru viz Příloha B.

5) **MySQL Enterprise Support:** poskytuje zejména nástroje pro vizualizaci dat. Tato funkcionalita není obsažena v Community Edition.

Tato práce se týká modelování dat a tvorby databáze. Čtenář bude seznámen pouze s omezenou funkcionalitou tohoto nástroje tak, aby byl schopen porozumět popisu jednotlivých způsobů tvorby fyzického datového modelu.

Samotné manipulaci s nástrojem MySQL Workbench předchází jeho instalace a základní konfigurace. Tam řadíme i vytvoření instance připojení k MySQL serveru a založení uživatelského účtu. Základní kroky, které je třeba učinit, jsou podrobněji popsány v Příloze B.

3.1 Úvodní stránka

Úvodní stránka je zobrazena vždy po spuštění aplikace. Skládá se ze tří částí: *MySQL Connections, Models, Shotrcuts*. Také obsahuje panel s položkami, které tvoří jakési hlavní menu aplikace (*File, Edit, View, Database, Tools, Scripting, Help*).

3.1.1 Shortcuts

V části *Shortcuts* lze nalézt odkazy na stránky s užitečnými tipy a podporou. Je tam například odkaz na dokumentaci k MySQL Workbench, různá fóra a blogy týkající se tohoto nástroje či seznam častých otázek a odpovědí.

3.1.2 MySQL Connections

Tato sekce obsahuje seznam instancí připojení k MySQL serveru. Jednotlivé instance obsahují základní informace, například název, uživatelský účet a schéma – pokud jsou s některými spojeny – či adresu a port pro navázání spojení se serverem. Je možné také vytvořit nové připojení či konfigurovat již stávající. K tomu slouží tlačítka s ikonami nacházející se pod nadpisem sekce.

K serveru se lze připojit dvojklikem na příslušnou instanci a zadáním správného hesla. Podrobnější popis tvorby instance viz Příloha B.

3.1.3 Models

Oddíl *Models* obsahuje seznam naposledy otevřených modelů a tři tlačítka. Tlačítko s ikonou malého plus pro vytvoření nového modelu, tlačítko s ikonou složky pro otevření již existujícího a tlačítko s šipkou, které nabízí možnost vytvoření modelu z fyzické databáze či skriptu.

3.2 Logický datový model - formulář

Tato stránka (viz obr. 1) je zobrazena jako nová záložka v případě vytváření nového či otevření již existujícího modelu. Stránka se skládá z hlavního menu, nástrojové lišty, a čtyř hlavních panelů: *Description Editor, User Types List/History panel, Model Overview, Modeling Additions*. Tyto panely lze zobrazit či skrýt pomocí tlačítek vedle pole pro vyhledávání.

3.2.1 Description Editor

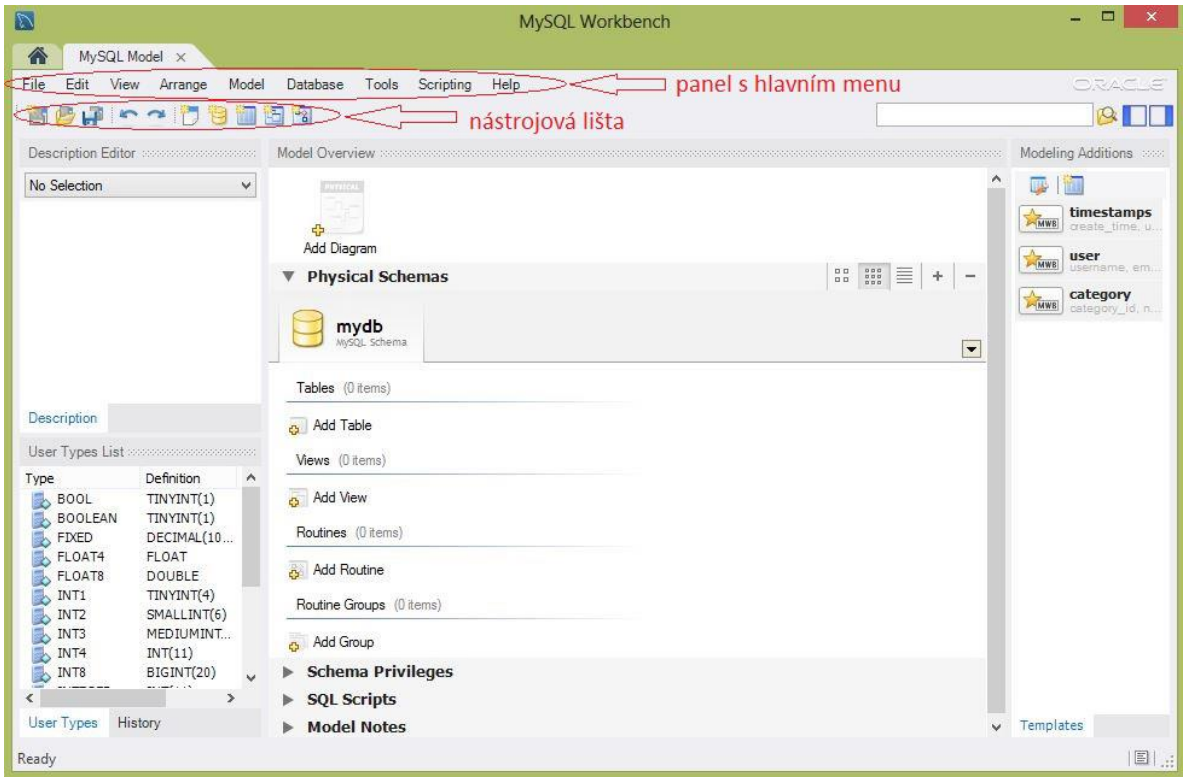
Description Editor slouží k přidávání popisu jednotlivým databázovým objektům. Nejprve vybereme objekt a do tohoto editoru pak napíšeme poznámky či popis vybraného objektu.

3.2.2 Model Overview

Tento panel obsahuje seznam vytvořených EER diagramů, databázových objektů, uživatelů a rolí. Dále pak umožňuje vkládat k modelu poznámky či vytvářet a zobrazovat přiložené SQL skripty.

V horní části panelu *Model Overview* je seznam vytvořených EER diagramů. Pokud již nějaký existuje, zobrazíme jej dvojklikem na příslušné tlačítko s ikonou označující daný diagram. Nový diagram lze vytvořit dvojklikem na *Add Diagram*, čímž je otevřena nová záložka s editorem (viz kapitola 3.3).

Další částí tohoto panelu je *Physical Schemas*, která obsahuje seznam otevřených schémat a jejich objektů. Seznam je možné zobrazit či skrýt pomocí šipky. Záhlaví této sekce obsahuje několik tlačítek. První tři umožňují nastavení způsobu zobrazení prvků



Obrázek 1: Stránka pro tvorbu logického datového modelu

schématu (například malé či velké ikony, seznam). Poslední dvě tlačítka (ikony plus a mínus) umožňují vytvořit nové či smazat otevřené schéma.

Tato sekce je dále rozdělena dle objektů na: *Tables, Views* (pohledy), *Routines, Routine Groups*. Jednotlivé objekty lze přidávat pomocí ikon (*Add Table, Add View*,…). Vytvořené prvky lze také přejmenovat, modifikovat, kopírovat či smazat. Tyto možnosti zobrazíme kliknutím pravého tlačítka myši na daný objekt.

Sekce *Schema Privileges* umožňuje vytvářet a modifikovat uživatele pro práci s jednotlivými schématy a přiřazovat jim práva prostřednictvím rolí (například právo na čtení/modifikaci dat v tabulkách apod.). Dvojklikem lze otevřít příslušný formulář. Nejprve je třeba definovat role, které pak přiřazujeme uživatelům. Při vytváření rolí je nutné definovat kromě názvu (případně rodiče) také práva k jednotlivým objektům – záložka *Privileges* na příslušném formuláři.

Formulář pro tvorbu a editaci objektů

Dvojklikem na libovolný databázový objekt je v dolní části obrazovky otevřen příslušný formulář. Používá se pro definování objektů a jejich modifikaci. Jednotlivé formuláře

jsou zobrazeny formou záložek, jejichž záhlaví obsahuje název modifikovaného objektu. Formulář pro tvorbu tabulek je popsán v kapitolách 4.1.1 a 4.2. Do ostatních formulářů doplníme název a SQL kód pro definování konkrétního objektu.

3.2.3 Hlavní menu

Menu *Tools* a *Scripting* obsahují funkce, které nejsou pro tuto práci potřebné. Nebudou proto v tomto textu popsány.

File

Menu *File* obsahuje volby pro vytvoření, otevření či uložení modelu. V případě otevření již existujícího je zobrazeno dialogové okno pro výběr uloženého modelu. Ten má koncovku *mwb*. Chceme-li zobrazit naposledy otevřené modely, zvolíme volbu *Open Recent*. Pro zavření aktuálně otevřeného modelu či EER diagramu zvolíme *Close Tab*. Totéž lze provést pomocí křížku na dané záložce.

Model můžeme uložit prostřednictvím položek *Save Model* či *Save Model As*. Po uložení modelu je jeho název zobrazen v záhlaví záložky. Modifikovali-li jsme model a chceme ho zavřít, aplikace nabídne možnost uložit změny. Model je uložen do souboru s koncovkou *mwb*. Databáze lze vytvářet nejen prostřednictvím tlačítek a formulářů, ale také importem. Postup je popsán v kapitole 4.3. Volba *Page Setup* umožňuje měnit orientaci, rozměry či okraje stránky. Stránku obsahující EER model pak lze vytisknout prostřednictvím volby *Print*.

Edit

Menu *Edit* umožňuje modifikovat objekty. Je možné je nejen modifikovat, ale i kopírovat, odebírat či přesunovat. Lze také vracet či opakovat naposledy provedené akce. Modifikovat můžeme nejen samostatné objekty, ale celé skupiny objektů, prostřednictvím *Edit Selected* (či *Edit Selected in New Window* – v novém okně). Abychom mohli objekty hromadně upravovat (či mazat), musíme je nejprve vybrat. K tomu slouží volba *Select*. Lze vybírat jak všechny objekty (*Select All*), tak pouze některé (například *Similar Figures* – nalezne a označí objekty podobné právě vybranému). Máme-li vybráno více objektů, můžeme se mezi nimi přepínat prostřednictvím *Go to Next/Previous Selected*. S hromadným výběrem se změní

i některé položky v menu *Edit* – konkrétně *Cut*, *Copy* a *Delete* budou v názvu obsahovat i počet vybraných objektů.

View

Menu *View* umožňuje zobrazit úvodní stránku (*Home*) či konzoli s výstupem (*Output*). Dále lze vybrat panely, které budou otevřeny (*Windows*). Je možné zvolit například *Zoom*, označení objektu (*Set Marker*) apod.

Menu *Arrange* se týká grafického návrhu databáze, je proto popsáno v kapitole 3.3.1.

Model

Toto menu obsahuje položky pro tvorbu diagramu (*Add Diagram*, *Create Diagram from Catalog Objects*). Další možnosti se týkají grafického návrhu, jsou proto popsány v kapitole 3.3.1.

Database

Menu *Database* umožňuje především navázání spojení s MySQL serverem (*Connect to Database*, *Manage Connections*). Dalšími možnostmi je například tvorba databáze z modelu (viz kapitola 4.1.3), tvorba modelu z databáze (viz kapitola 3.5.1) či migrace dat z jiného databázového systému (viz kapitola 4.3.2). Volba *Compare Schemas* zajistí porovnání dvou schémat. Případné rozdíly pak vypíše v reportu. Volba *Synchronize model* zajistí synchronizaci modelu s libovolnou existující databází.

Help

Program nabízí standardní nápovědu ke všem funkcionalitám, které aplikace nabízí. Další nabízenou možností je propojení s webovými stránkami *www.mysql.com*, kde lze vyhledat potřebnou pomoc. Menu *Help* dále obsahuje volby pro zobrazení logů či systémových informací.

3.2.4 Nástrojová lišta

Nástrojová lišta obsahuje tyto tlačítka:

- *New Document* – vytvoření nového modelu
- *Open Model from File* – otevření existujícího modelu

- *Save Model to Current File* – uložení právě otevřeného modelu
- *Undo/Redo* – vrácení či opakování právě provedené akce
- *Add new Diagram* – vytvoření nového EER diagramu
- *Add new Schema* – založení nového schématu
- *Add new Table* – založení nové tabulky
- *Add new View* – založení nového pohledu
- *Add new Routine* – přidání nové funkce

3.2.5 User Types list/History

V této sekci je možné editovat vlastnosti datových typů, které lze následně přiřazovat atributům. Záložka *History* pak obsahuje nedávno provedené akce.

3.2.6 Modeling Additions

Tento panel obsahuje šablony tabulek. Chceme-li vytvářet tabulky, které budou mít podobné atributy nebo alespoň jejich část, je výhodné vytvořit šablonu. Kliknutím na první tlačítko zleva – *Open the table template editor* – je otevřeno okno se seznamem šablon a formulářem pro jejich editaci. Tlačítkem *New Template* vytvoříme novou šablonu. Její název můžeme libovolně změnit.

Dále je možné definovat atributy, jejich datové typy, výchozí hodnoty i integritní omezení (*Column Name, Datatype, Default, PK, NN, UQ, AI*). Více o datových typech a integritním omezení viz kapitoly 2.2, 2.3 a 2.4. Aplikace nabízí i možnost definice znakové sady (*Column Collation*), je však doporučeno nastavit jednotné kódování pro celou databázi, ne pro jednotlivé tabulky či sloupce.

Máme-li definováno vše potřebné, můžeme okno zavřít. Novou tabulku pak vytvoříme dvojklikem na konkrétní šablonu v *Modeling Additions* či zvolením šablony a kliknutím na druhé tlačítko v tomto panelu (*Create a new table based on the selected table template*).

3.3 Logický datový model - EER Diagram

Tato stránka (viz obr. 2) je zobrazena jako záložka v případě tvorby EER diagramu. Skládá se z těchto částí: *Diagram, Bird's Eye, Catalog Tree / Layer Tree / User Types List, Description/Properties Editor, Modeling Additions*. Panel *Modeling Additions* je popsán výše.

3.3.1 Hlavní menu

Panel s položkami hlavního menu je stejný jako na stránce pro tvorbu logického datového modelu popsané výše.

Model

Kromě tvorby diagramu obsahuje tato položka také možnost měnit vzhled databázových objektů dle definovaných vzorů (*Object Notation, Relationship Notation*). Menu *Model* také umožňuje například změnu velikosti diagramu či jeho umístění v poli (*Diagram Properties and Size*).

Arrange

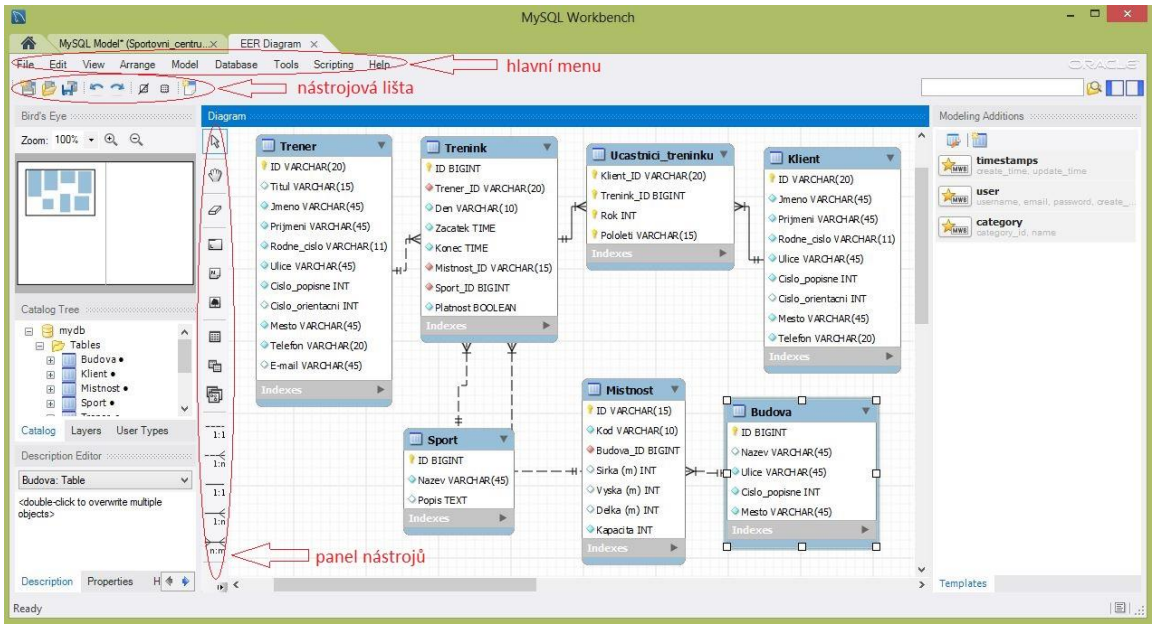
Prostřednictvím položky *Arrange* lze nastavit vlastnosti grafického návrhu modelovaného systému. Jedná se například o přenesení některých databázových objektů do popředí či do pozadí.

3.3.2 Nástrojová lišta

Většina tlačítek v nástrojové liště je již popsána v předešlé kapitole týkající se logického datového modelu – formuláře. Lišta navíc obsahuje například tlačítko pro vytvoření nového diagramu.

3.3.3 Diagram

Panel *Diagram* obsahuje pole pro umístění databázových objektů a panel nástrojů s tlačítky pro jejich tvorbu. Tvorba objektů i relací je popsána v kapitole 4.1.1.



Obrázek 2: Stránka pro tvorbu diagramu

3.3.4 Bird's Eye

Panel *Bird's Eye* poskytuje náhled na všechny objekty umístěné v diagramu. Tato vlastnost se hodí především pro diagramy obsahující větší množství tabulek. Diagram je možné přiblížit či oddálit (zoom).

3.3.5 Catalog Tree / Layer Tree / User Types List

Záložka *Catalog Tree* obsahuje všechna otevřená schémata a jejich objekty. Slouží především k přidávání vytvořených objektů do diagramu (drag and drop). V záložce *Layer Tree* najdeme seznam tabulek v modelu a jejich cizí klíče. Záložka *User Types List* umožňuje editovat vlastnosti datových typů, které lze následně přiřazovat atributům.

3.3.6 Description / Properties Editor

Description Editor slouží k přidávání popisu jednotlivým databázovým objektům. Nejprve vybereme objekt a do tohoto editoru pak napíšeme poznámky či popis vybraného objektu.

Zajímavá je záložka *Properties*, kde je možné měnit vzhled grafického zobrazení databázových objektů. Lze měnit například jejich barvu či velikost.

3.4 Fyzický datový model

Stránka pro tvorbu fyzického datového modelu (viz obr. 3) je otevřena formou nové záložky. Její zobrazení je podmíněno připojením k MySQL serveru. Na úvodní stránce proto zvolíme konkrétní instanci a zadáme heslo.

Stránka obsahuje hlavní menu, nástrojovou lištu, SQL editor a panely: *Navigator, Information, Output, SQL Additions*. SQL editor je blíže popsán v kapitole 4.4, která pojednává o jednom ze způsobů tvorby databáze. Panel *Information* obsahuje informace o vybraném databázovém objektu či o daném připojení k serveru MySQL.

3.4.1 SQL Additions

SQL Additions obsahuje záložku *Snippets*, kde najdeme seznam snippetů (tj. znovupoužitelných kusů kódu). Máme-li v SQL editoru napsaný kód, který budeme používat vícekrát, uložíme si jej jako snippet pomocí tlačítka s ikonou hvězdičky a malého plus v panelu *SQL Additions*. V editoru jej pak zobrazíme dvojklikem na příslušný snippet. Další záložkou je kontextová nápověda, která napovídá správnou syntaxi SQL příkazů při psaní kódu do SQL editoru.

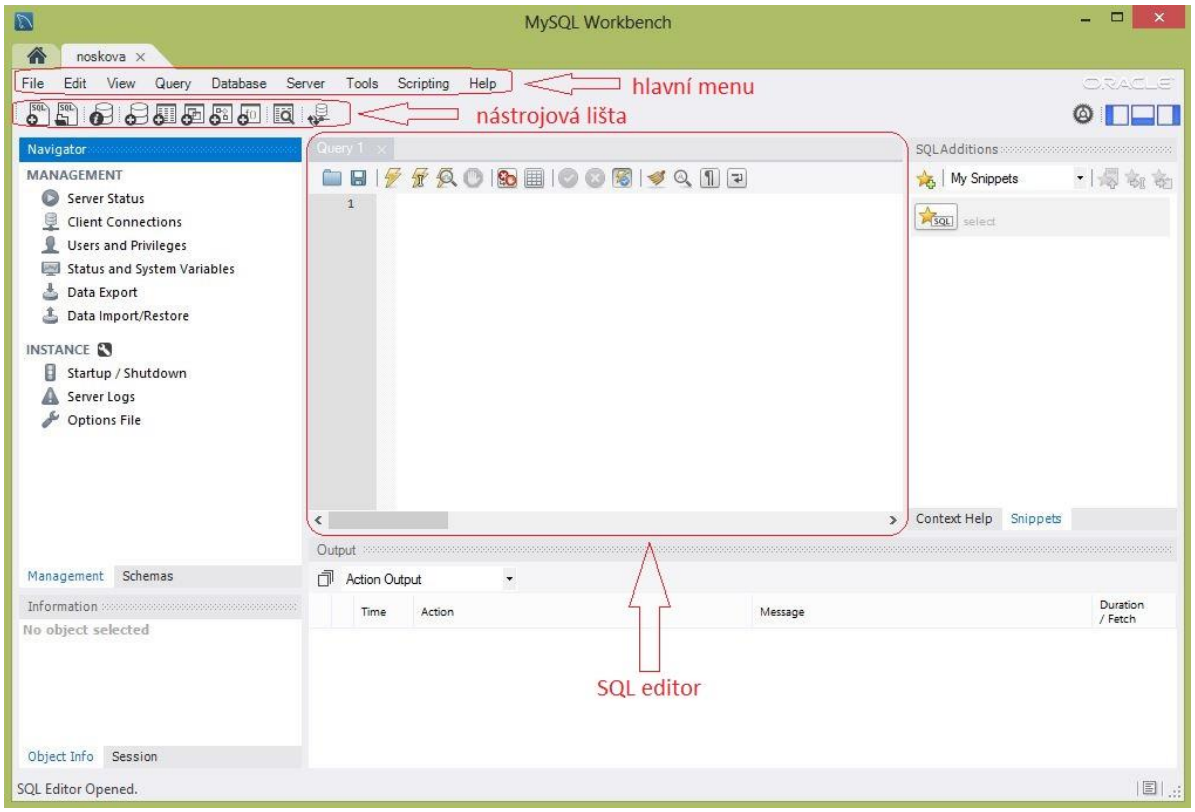
3.4.2 Navigator

Tento panel obsahuje dvě záložky. V záložce *Schemas* jsou zobrazena všechna existující schémata i jejich databázové objekty. Jednotlivé objekty lze zobrazit či skrýt pomocí šipek. Každá databáze může obsahovat tabulky, pohledy, funkce a procedury. Tvorbě tabulek je věnována kapitola 4.2, která popisuje jeden ze způsobů tvorby databáze.

Druhou záložkou panelu *Navigator* je *Management*. Slouží pro administraci a konfiguraci serveru, správu uživatelů či import/export dat. Import je popsán v kapitole 4.3, export v kapitole 4.1.2.

3.4.3 Output

Panel *Output* slouží pro komunikaci s uživatelem. Informuje ho o úspěšných i neúspěšných výstupech prováděných akcí. Výstupy lze filtrovat.



Obrázek 3: Stránka pro tvorbu fyzického datového modelu

Nastane-li problém a tudíž potřeba kontaktovat administrátora, je vhodné sdělit mu znění případné chybové hlášky v tomto panelu.

3.5 Migrace databáze

Databázi či pouze samotné databázové objekty lze exportovat do souboru. Taktéž i data. Postup pro export je popsán v kapitole 4.1.2. Co se týče vlastní migrace databáze, je možné převést logický datový model na databázi (postup popsán v kapitole 4.1.3) či naopak.

3.5.1 Převod databáze do logického datového modelu

Máme-li vytvořenou databázi a chceme vidět propojení tabulek prostřednictvím grafického modelu, můžeme fyzický datový model převést na logický datový model.

V hlavním menu vybereme *Database -> Reverse Engineer*. Zobrazí se okno, kde zvolíme konkrétní instanci připojení k MySQL serveru. Další stránka je věnována výsledku autentizace (tj. zda bylo či nebylo zadáno správné heslo).

Vybereme schéma či schémata, která chceme migrovat. Následuje opět stránka s výsledkem autentizace. Dále je vypsán seznam databázových objektů, která zvolená schémata obsahují. K migraci je možné zvolit pouze některé z nich.

Zaškrtneme-li volbu *Place objects on diagram*, budou všechny migrované objekty umístěny do pole pro grafické znázornění modelu. V opačném případě budou pouze v seznamu objektů vygenerovaného logického modelu. Celý proces migrace dokončíme kliknutím na tlačítko *Finish*. Vygenerovaný model je otevřen a zobrazen prostřednictvím nové záložky.

3.6 Editace dat

Máme-li vytvořenou databázi, tedy všechny tabulky i vazby, můžeme začít vkládat data. I pro tento proces existuje více postupů. První z nich je psaní SQL příkazů do editoru. Pro vkládání dat se používá příkaz `INSERT`. Proces dokončíme kliknutím na tlačítko  či volbou *Query->Execute* v hlavním menu. Dalším způsobem vkládání dat do databáze je kliknutí pravého tlačítka myši na tabulku, ve které chceme editovat data, a zvolením *Select Rows*. Tím je otevřen formulář pro editaci dat. Součástí formuláře je nástrojová lišta a data, které tabulka již obsahuje.

MySQL Workbench, stejně jako jiné databázové nástroje, nabízí různé možnosti manipulace s daty. Můžeme je přidávat, mazat, upravovat, dokonce i exportovat či importovat do/ze souboru. Všechny tyto funkce zajišťují tlačítka v nástrojové liště.

Hodnoty pro atributy, které mají nastavenou vlastnost `AUTO_INCREMENT`, nevyplňujeme. K jejich automatickému vyplnění dojde až poté, co klikneme na tlačítko *Apply*.

3.7 Modifikace tabulek

Může se vyskytnout potřeba změnit již existující tabulky. Nemusí jít jen o jejich název, ale například o přidání či odebrání atributů, referencí apod. V takových případech klikneme pravým tlačítkem myši na tabulku, které se změna týká, a zvolíme *Alter table…*. Zobrazí se vyplněný formulář – stejný jako pro tvorbu tabulky nové, vytvářeli-li jsme databázi nezávisle na logickém datovém modelu. Práce s tímto formulářem je podrobně popsána v kapitole 4.2.

4 Způsoby realizace databáze

V této kapitole jsou popsány čtyři způsoby, kterými lze vytvořit databázi pomocí nástroje MySQL Workbench. Konkrétně se jedná o její vygenerování z logického datového modelu, tvorbu databáze pomocí formulářů a funkcí nástroje MySQL Workbench, import ze souboru či tvorbu kódu v SQL editoru. Ke každé z uvedených cest se váže několik problémů, které mohou nastat. Jejich popis i řešení je uvedeno na konci každé z podkapitol. Protože jednotlivé cesty na sobě nejsou závislé a čtenář může zvolit jakoukoli z nich, mohou se některé odstavce textu v následujících kapitolách opakovat.

Při popisu těchto způsobů jsem vycházela jednak z vlastních poznatků, ale také z manuálu k nástroji MySQL Workbench [4].

Před samotnou tvorbou databáze je samozřejmostí spuštění aplikace a připojení k MySQL serveru[3]. Po připojení je otevřena záložka reprezentující konkrétní instanci. Seznam schémat a databází, jež patří k jednotlivým instancím, obsahuje záložka *Schemas* v oddíle *Navigator*. Při tvorbě databází je nutné začínat těmi tabulkami, které neobsahují cizí klíče. Teprve po definování příslušných atributů na ně můžeme vytvořit referenci v jiné tabulce.

4.1 Tabulky vygenerované z logického modelu

Tato cesta navazuje na tvorbu logického datového modelu čili návrhu architektury databáze bez jejího fyzického uložení. Podrobnější popis rozdílu mezi logickým datovým modelem a databází nalezneme v kapitole 2.1.

Tvorba databáze je založena na jejím grafickém návrhu. Uživatel vidí všechny tabulky i jejich vzájemné vazby. Návrh lze snadno a rychle modifikovat, tj. měnit vazby mezi tabulkami, přidávat či odebírat databázové objekty nebo měnit jejich uspořádání.

Jakmile docílíme takové podoby návrhu, kdy model odpovídá skutečnosti, necháme jej pomocí několika kroků automaticky převést do fyzického datového modelu.

[3] Jestliže neběží MySQL server, je nutné jej spustit. Postup pro spuštění serveru je popsán v Příloze B.

4.1.1 Logický datový model

Prvním krokem tohoto procesu tvorby databáze je vytvoření jejího návrhu. Logický datový model je vlastně přehlednější grafická náhrada ručně kresleného modelu. Tento model je možné vytvořit dvěma způsoby, které jsou popsány v kapitolách níže. Jedná se buď o vytváření objektů v poli či nadefinování tabulek a jejich převedení do diagramu. Výsledkem bude téměř stejný logický datový model. Jediným rozdílem budou názvy cizích klíčů a rozkladových tabulek, které se v případě vytváření objektů v poli generují automaticky, zatímco u převádění definovaných tabulek do diagramu je zakládá uživatel.

Logický datový model může vytvořit jakýkoli uživatel MySQL Workbench. Nemusí mít založený účet ani být připojený k MySQL serveru.

Vytváření objektů v poli

Jedná se o způsob vytvoření logického datového modelu prostřednictvím přidávání databázových objektů do pole a jejich následnou modifikací.

Na úvodní obrazovce programu MySQL Workbench zvolíme ikonu malého plus v části *Models* či volbu *File -> New Model*. Tím se otevře záložka pro vytvoření modelu, který budeme vytvářet v rámci schématu. Schéma libovolně pojmenujeme. Formulář pro definování nového názvu či přiřazení znakové sady zobrazíme dvojklikem na příslušné schéma. Nastavíme takovou znakovou sadu, jež bude použita u všech databázových objektů, které vytvoříme. V současné době se nejčastěji používá UTF-8.

Dvojklikem na *Add Diagram* se otevře záložka s polem, kam budeme umisťovat tabulky. Po levé straně tohoto pole je nabídka objektů, které lze vytvořit. Do pole můžeme umisťovat nejen tabulky a relace, ale například i obrázky či textová pole. Doporučuji začít tabulkami.

Tabulku vytvoříme kliknutím na ikonu  a umístíme ji na libovolné místo v poli. Její název je vygenerován – vždy `tableX`, kde X je celé číslo, nebo jinak, dle základního nastavení programu. Dvojklikem na tabulku je zobrazen formulář pro definování jejích vlastností. Je dobré definovat především název (*Table Name*), atributy a jejich datové typy, integritní omezení či výchozí hodnoty (*Column Name, Datatype, PK, NN, UQ, BIN, UN, ZF, AI, Default*). Název tabulky by neměl obsahovat mezery (možno nahradit

podtržítky) ani diakritiku. Více informací o datových typech a integritním omezení viz kapitoly 2.2, 2.3 a 2.4.

Vedle položky *Schema,* které označuje schéma, v němž je databáze tvořena, je ikona zdvojené šipky. Ta obsahuje další položky, které je možné definovat. Jsou to především *Collation* (znaková sada) a *Engine* (typ tabulky). Přestože to není povinné, znaková sada by měla být stejná pro celou databázi. Co se týče enginů, nejpoužívanějšími typy tabulek jsou InnoDB a MyISAM. Chceme-li zajistit referenční integritu (čili propojení tabulek cizími klíči), měli bychom zvolit InnoDB. Více o typech enginů viz kapitola 2.6.

Budeme-li vytvářet podobné tabulky, je výhodné použít šablonu. Nástroj MySQL Workbench nabízí tři předdefinované šablony, zároveň však umožňuje vytvořit další. Šablony najdeme v oddíle *Modeling Additions*. Tabulku vytvoříme dvojklikem na příslušnou šablonu. Tvorbu šablony popisuje kapitola 3.2.6.

Při vytváření modelu nejprve vytvoříme jednotlivé tabulky. Jakmile jsou všechny vytvořené, můžeme je propojit pomocí relací. Pro vytvoření relace musí mít alespoň jedna z tabulek min 1 atribut. Zvolíme vhodnou relaci. Při výběru dbáme na kardinalitu vztahu, povinnost vazby je možné řešit prostřednictvím formuláře, který je popsán v textu níže. Dále rozlišujeme, zda jde o *Identifying* či *Non-Identifying* vazbu (typy vazeb viz kapitola 2.1.1). Do tabulky, na kterou pak klikneme nejdříve, budou automaticky přidány všechny atributy, které jsou součástí primárního klíče druhé tabulky. Je tedy třeba si uvědomit, do které tabulky chceme přidat referenci. Toto však neplatí pro relaci M:N. V takovém případě se nevytvoří sloupec s cizím klíčem, ale nová rozkladová tabulka. Vytvořené relace je možné upravovat pomocí formuláře. Zobrazíme jej dvojklikem na příslušnou relaci. Ve formuláři lze definovat titulek/název vztahu mezi tabulkami z obou směrů. Například relaci mezi entitami `Místnost` a `Budova` lze pojmenovat následovně: místnost **se nachází** v budově vs. budova **obsahuje** místnosti. Důležitější je však záložka *Foreign Keys*, kde můžeme změnit kardinalitu vazby, zvolit zda je či není *Identifying* a z které strany je či není povinná (*Mandatory*).

Posledním krokem může být úprava vygenerovaných názvů atributů cizího klíče či názvu samotného cizího klíče (formulář pro tvorbu tabulek, záložka *Foreign Keys* atribut *Foreign Key Name*).

Převedení definovaných tabulek do diagramu

Jedná se o definování tabulek a jejich atributů. Tento způsob nezačíná grafickým znázorněním databáze jako kapitola výše. V tomto případě je třeba tabulky nejprve vytvořit, pak je lze převést do grafického modelu.

Na úvodní obrazovce programu MySQL Workbench zvolíme ikonu malého plus v části *Models* či volbou *File -> New Model*. Tím se otevře záložka pro vytvoření modelu, který budeme vytvářet v rámci schématu. Schéma libovolně pojmenujeme. Formulář pro definování nového názvu či přiřazení znakové sady zobrazíme dvojklikem na příslušné schéma. Nastavíme takovou znakovou sadu, jež bude použita u všech databázových objektů, které vytvoříme. V současné době se nejčastěji používá UTF-8.

Tabulky či obecně databázové objekty vytvoříme buď pomocí tlačítek v nástrojové liště pod položkami hlavního menu (*File, Edit, View*, …) nebo pomocí tlačítek (*Add Table, Add View* apod) nacházejících se zhruba uprostřed stránky.

Ve spodní části stránky se zobrazí formulář pro definování vlastností tabulky. Je dobré definovat především název (*Table Name*), atributy a jejich datové typy, integritní omezení či výchozí hodnoty (*Column Name, Datatype, PK, NN, UQ, BIN, UN, ZF, AI, Default*). Název tabulky by neměl obsahovat mezery (možno nahradit podtržítky) ani diakritiku. Více informací o datových typech a integritním omezení viz kapitoly 2.2, 2.3 a 2.4.

Vedle položky *Schema,* která označuje schéma, v němž je databáze tvořena, je ikona zdvojené šipky. Ta obsahuje další položky. Především jsou to *Collation* (znaková sada) a *Engine* (typ tabulky). Přestože to není povinné, znaková sada by měla být stejná pro celou databázi. Nejpoužívanějšími typy tabulek jsou InnoDB a MyISAM. Chceme-li zajistit referenční integritu (čili propojení tabulek cizími klíči), měli bychom zvolit InnoDB. Více o typech enginů viz kapitola 2.6.

Po vytvoření tabulek je musíme propojit relacemi. Toho docílíme definováním cizích klíčů. Nejprve je třeba založit atributy, které budou reprezentovat cizí klíč. V záložce *Foreign Keys* pak dokončíme tvorbu referenční integrity definováním názvu cizího

klíče (*Foreign Key Name*), zvolením tabulky, na kterou chceme odkazovat (*Referenced Table*), a nakonec samotné vytvoření reference. Tím posledním krokem je myšlena volba sloupců z právě editované tabulky (*Column*) a volba odpovídajících sloupců z jiné tabulky (*Referenced Column*).

Chceme-li vytvořit relaci M:N, musíme nejprve vytvořit rozkladovou tabulku. Problematikou složených vazeb se zabývá kapitola 2.1.1. Vytváříme-li rozkladovou tabulku, postupujeme stejně jako u kterékoli jiné.

Po vytvoření všech tabulek a jejich vzájemném propojení je možné zobrazit jejich grafické rozložení. Zvolíme tedy *Add Diagram*. Otevře se nová záložka s prázdným polem. V oddílu *Catalog Tree* je seznam všech vytvořených objektů vztahujících se k danému schématu. Myší postupně přetáhneme jednotlivé tabulky do pole. Chceme-li zobrazit pouze část modelovaného systému, vybereme pouze některé tabulky. Na základě definovaných cizích klíčů jsou vytvořeny relace, které spolu s tabulkami představují grafický návrh databáze.

Další možností je namísto *Add Diagram* zvolit v hlavním menu *Model* volbu *Create Model from Catalog Objects*. Tím dojde k automatickému vygenerování grafického návrhu skládajícího se ze všech vytvořených databázových objektů.

Vytvořené relace je možné upravovat pomocí formuláře, který zobrazíme dvojklikem na příslušnou relaci. Více informací o formuláři lze najít v popisu prvního způsobu tvorby logického datového modelu. Nástroj MySQL Workbench neumí umístit tabulky do pole tak, aby se relace nekřížily. Je proto třeba jednotlivé objekty vhodně rozmístit tak, aby byl diagram přehledný.

4.1.2 Vygenerování databáze do skriptu

Z navrženého logického datového modelu je možné vygenerovat SQL kód pro tvorbu fyzického datového modelu. Kód lze uložit (exportovat) do souboru.

V záložce *File* vybereme možnost *Export* a *Forward Engineer SQL CREATE Script…* . Otevře se okno, v jehož horní části můžeme zvolit vhodný název skriptu, chceme-li vygenerovaný kód uložit do souboru, a adresář pro jeho umístění.

Na další stránce lze zaškrtnout nějaké z nabízených možností. Doporučuji vybrat možnost *Omit Schema Qualifier in Object Names*. Kdybychom to neudělali, vytvořené

a exportované objekty by ve svých názvech obsahovaly i název schématu, ve kterém byly vytvořeny. Tím by vznikl problém při pokusu o import do schématu s jiným názvem. Klikneme na tlačítko *Next* a zkontrolujeme, zda se do skriptu ukládá opravdu všechno. Jestliže je vše v pořádku, opět klikneme na *Next*. Ukáže se nám vygenerovaný kód v jazyce SQL. Proces dokončíme kliknutím na tlačítko *Finish*.

Tímto získáme soubor s koncovkou *sql*, který lze použít například jako zdroj importovaných dat do jiného databázového systému.

4.1.3 Převedení logického datového modelu do fyzické podoby

Začínáme v okamžiku, kdy máme hotový logický datový model (možnosti postupu popisují předchozí kapitoly).

V hlavním menu zvolíme možnost *Database* a poté *Forward Engineer…* . Zobrazí se okno, kde definujeme, k jakému připojení a do jakého schématu bude model generován. Na další stránce můžeme zvolit některé z nabízených kroků. Doporučuji zaškrtnout možnost *Omit Schema Qualifier in Object Names*. Kdybychom to neudělali, vytvořené a exportované objekty by ve svých názvech obsahovaly i název schématu, ve kterém byly vytvořeny. Tím by vznikl problém při pokusu o import do schématu s jiným jménem.

Pro další krok potřebujeme být připojení k serveru. Nejsme-li připojeni, aplikace požádá o heslo a připojí se k serveru. Po autentizaci nástroj MySQL Workbench zobrazí seznam databázových objektů, které budou vytvořeny, přičemž uživatel může vybrat, které chce vytvořit a které ne. Na další stránce jsou zobrazeny SQL příkazy, jež budou provedeny. Posledním krokem je provedení SQL příkazů a vytvoření databáze i databázových objektů. Poslední stránka obsahuje report procesu či případné chyby a varování. Některé z chyb jsou popsány v kapitole níže. Pokud vše proběhlo v pořádku, můžeme okno zavřít.

V *Navigatoru* v záložce *Schemas* je pak v rámci příslušného schématu vytvořena databáze, kterou můžeme plnit daty (viz kapitola 3.6).

4.1.4 Možné problémy a jejich řešení

1) **Problém:** Při definování cizího klíče nelze zaškrtnout některý(é) z atributů, které jej budou tvořit.

   Řešení: Sloupce z tabulky, v níž vytváříme cizí klíč a které jsou součástí cizího klíče, musí být stejného datového typu jako sloupce, na něž vytváříme referenci.

2) **Problém:** Převod logického datového modelu do fyzické podoby neproběhl – vyskytly se chyby.

   Řešení: Je nutné pročíst logy na poslední obrazovce okna *Forward Engineer*. Příčinou mohou být například nedostatečná práva.

Při tvorbě logického datového modelu aplikace dovolí vytvořit tabulky se stejným názvem, což je problém při převodu do fyzické podoby. Proces sice proběhne, ale bude vytvořena pouze jedna z těchto tabulek.

4.2 Databáze vytvořená nezávisle na logickém datovém modelu

Touto cestou lze vytvořit fyzický datový model bez logického návrhu. Není tedy použito grafické podoby modelu.

V oddílu *Navigator* zobrazíme záložku *Schemas*. Ta obsahuje seznam databází, které jsou vytvořené v rámci založených schémat. Ke každému schématu lze zobrazit seznam objektů (konkrétně tabulky, funkce, pohledy, procedury), které mu přísluší, i jejich vlastnosti.

Nejprve je třeba vytvořit prázdné schéma. Založení nového schématu se skládá z několika jednoduchých kroků. Po připojení k serveru je třeba nejprve kliknout na ikonu  v horní nástrojové liště. Otevře se nová záložka pro definování názvu schématu (je doporučeno zadávat pouze alfanumerické znaky a mezery nahradit podtržítky) a zvolení znakové sady. Přestože to není povinné, znaková sada by měla být stejná pro celou databázi. V současnosti se nejčastěji používá UTF-8. Po kliknutí na *Apply* se zobrazí okno s příkazem pro vytvoření schématu v jazyce SQL. Příkaz sice můžeme modifikovat, avšak nedoporučuje se to. Celý proces tvorby je dokončen opět tlačítkem *Apply*. Jestliže vše proběhlo v pořádku, nové prázdné schéma se objeví v seznamu schémat v *Navigatoru*.

Zobrazíme vlastnosti vytvořeného schématu a kliknutím pravého tlačítka na *Tables* zobrazíme možnosti. Zvolíme *Create table*. Dojde k zobrazení záložky s formulářem pro definování nové tabulky. Je možné otevřít více těchto záložek najednou, tedy vytvářet či editovat více tabulek současně. Formulář se skládá z několika částí, největší prioritu však mají záložky *Columns,* v případě propojování tabulek i záložka *Foreign Keys*.

Základem tvorby tabulky je definice jejího názvu (*Table Name*), atributů a jejich datových typů, integritního omezení či výchozích hodnot (*Column Name, Datatype, PK, NN, UQ, BIN, UN, ZF, AI, Default*) – více o datových typech a typech integritního omezení viz kapitoly 2.2, 2.3 a 2.4. Název tabulky by neměl obsahovat mezery (možno nahradit podtržítky) ani diakritiku, stejně tak názvy sloupců. S jednotlivými atributy lze dále manipulovat, například je modifikovat, mazat či měnit jejich pořadí. Ve sloupci *Default* je možné nastavit výchozí hodnotu atributu.

Pod tabulkou se seznamem atributů je formulář, který poskytuje nejen větší přehled o jednotlivých vlastnostech, ale také umožňuje definovat sloupcům více vlastností. Jde zejména o znakovou sadu (*Collation*). Avšak u této vlastnosti je lépe ponechat stejné nastavení pro celou tabulku či lépe – pro celou databázi.

Vedle položky *Schema* je tlačítko s ikonou zdvojené šipky. Pod ní se skrývají další vlastnosti, které je možné zvolit. Konkrétně se jedná o znakovou sadu (*Collation*) pro celou tabulku a engine neboli typ tabulky. Nejčastěji používanými tabulkami jsou InnoDB a MyISAM. Chceme-li zajistit referenční integritu (čili propojení tabulek cizími klíči), měli bychom zvolit InnoDB. Více o typech enginů viz kapitola 2.6.

Jak je zmíněno v úvodu kapitoly čtyři, doporučuje se začít tabulkami, které nemají cizí klíče. Do těch, které obsahují cizí klíč, pak přidáme sloupce, které budou jeho součástí. V záložce *Foreign Keys* dokončíme tvorbu referenční integrity definováním názvu cizího klíče (*Foreign Key Name*), zvolením tabulky, na kterou chceme odkazovat (*Referenced Table*), a nakonec samotné vytvoření reference. Tím posledním krokem je myšlena volba sloupců z právě editované tabulky (*Column*) a volba odpovídajících sloupců z jiné tabulky (*Referenced Column*).

Kromě atributů a cizích klíčů je samozřejmě možné definovat například indexy, které napomáhají rychlejšímu prohledávání tabulek. Pro dokončení procesu tvorby tabulky

klikneme na *Apply*. Zobrazí se okno s kódem v jazyce SQL pro vytvoření tabulky. Tlačítkem *Apply* pak dojde k aplikaci skriptu, tedy k fyzickému vytvoření tabulky v databázi. Celý proces je dokončen tlačítkem *Finish*.

Jestliže vytváříme podobné tabulky, je výhodné vytvořit si šablonu. Tedy v *Navigatoru* v záložce *Schemas* zobrazíme vlastnosti databáze a kliknutím pravého tlačítka na *Tables* zobrazíme možnosti. Volbou *Create Table Like… -> Edit Templates…* je otevřeno okno (*Table Templates*) pro vytvoření nové šablony či modifikaci již existujících. Kliknutím na tlačítko *New Template* bude do seznamu šablon přidána nová. Její název můžeme libovolně změnit. Dále je možné definovat atributy, jejich datové typy, výchozí hodnoty i integritní omezení (*Column Name, Datatype, Default, PK, NN, UQ, AI*). Aplikace nabízí i možnost definice znakové sady (*Column Collation*), je však doporučeno nastavit jednotné kódování pro celou databázi, ne pro jednotlivé tabulky či sloupce.
Máme-li definováno vše potřebné, můžeme okno zavřít. Novou tabulku pak z této šablony vytvoříme kliknutím pravého tlačítka myši na *Tables*, zvolením možnosti *Create Table Like…* a vybráním konkrétní šablony.

Při tvorbě databáze tedy nejprve vytvoříme všechny tabulky (ať už použijeme šablony či ne) a reference. Během jejich definování lze totiž narazit na chybu v návrhu architektury, která se v databázi naplněné daty hůře opravuje. Data je doporučeno vkládat až po vytvoření celé databáze. Plnění a editace dat viz kapitola 3.6.

4.2.1 Možné problémy a jejich řešení

1) **Problém:** Při definování cizího klíče nelze zaškrtnout některý(é) z atributů, které jej budou tvořit.

   Řešení: Sloupce z tabulky, v níž vytváříme cizí klíč a které jsou součástí cizího klíče, musí být stejného datového typu jako sloupce, na něž vytváříme referenci.

2) **Problém:** Proces vytváření tabulky skončí s chybou.

   Řešení: Je nutné pročíst logy na poslední obrazovce okna, které je zobrazeno po kliknutí na *Apply*. Příčinou mohou být například nedostatečná práva.

4.3 Import ze souboru

Přestože je tento postup popsán jako způsob vytvoření databáze, lze jej použít i k obnovení databáze ze zálohy.

Při volbě této cesty se předpokládá, že uživatel má soubor vhodný k importu. Tedy soubor, který obsahuje SQL příkazy pro vytvoření celé databáze či jen samotných databázových objektů. Lze jej získat například exportem[4] z jiného databázového serveru – musí však jít o systém MySQL. Je možné importovat buď celou databázi, nebo pouze samotné databázové objekty. V takovém případě je dobré si předem založit schéma, kam budou objekty umístěny.

Založení nového schématu se skládá z několika jednoduchých kroků. Po připojení k serveru je třeba nejprve kliknout na ikonu  v horní nástrojové liště. Otevře se nová záložka pro definování názvu schématu (je doporučeno zadávat pouze alfanumerické znaky a mezery nahradit podtržítky) a zvolení znakové sady. Přestože to není povinné, znaková sada by měla být stejná pro celou databázi. V současnosti se nejčastěji používá UTF-8. Po kliknutí na *Apply* se zobrazí okno s příkazem pro vytvoření schématu v jazyce SQL. Celý proces tvorby je dokončen opět tlačítkem *Apply*. Jestliže vše proběhlo v pořádku, nové schéma se objeví v seznamu v *Navigatoru*.

Máme-li vhodný soubor, případně vytvořené schéma, můžeme začít importovat.

Nejprve zvolíme jednu z možností (viz obr. 4):

a) v hlavním menu vybereme *Server -> Data Import*,

b) v oddíle *Navigator* zvolíme záložku *Management -> Data Import/Restore*.

Aplikace bude opět požadovat zadání hesla. Po autentizaci se otevře nová záložka věnovaná nastavení importu dat.

Na základě způsobu uložení importovaných dat zvolíme buď *Import from Dump Project Folder* nebo *Import from Self-Contained File.*

Je-li celá databáze (všechny databázové objekty) uložena v jednom souboru, zvolíme možnost *Import from Self-Contained File* a vybereme jej. Pokud soubor obsahuje pouze

[4] MySQL Workbench taktéž umožňuje export databáze a databázových objektů do souboru. Postup pro export je popsán v kapitole 4.1.2.

databázové objekty ale ne databázi[5], musíme navíc vybrat schéma, kam budou objekty umístěny. V takovém případě je dobré mít založené prázdné schéma. Pokud tomu tak není, lze jej vytvořit i nyní – prostřednictvím tlačítka *New…*. Máme-li již předem založené schéma, vybereme jej v položce *Default Target Schema*. Nastane-li opačný problém, tj. soubor obsahuje příkazy pro vytvoření databáze a uživatel přesto zvolí schéma, není k tomu přihlíženo. Při generování importované databáze se automaticky vytvoří stejnojmenné schéma, které ji nahrazuje.

Jestliže jsou jednotlivé databázové objekty uloženy v samostatných souborech, zvolíme možnost *Import from Dump Project Folder*. Vybereme složku a zvolíme buď celou databázi či pouze jednotlivé objekty.

Klikneme na tlačítko *Start Import*. Tím dojde k provedení příkazů jazyka SQL, které jsou obsaženy v importovaném souboru, tedy k vytvoření fyzického datového modelu.

Databázi nyní můžeme naplnit daty (viz kapitola 3.6).

4.3.1 Možné problémy a jejich řešení

1) **Problém:** Nelze vytvořit definované schéma.

   Řešení: Zvolit jiný název schématu.

2) **Problém:** Zvolíme *Import from Self-Contained File* a spustíme import. Aplikace zahlásí, že vybraný soubor neexistuje.

3) **Řešení:** Při zvolení tohoto importu je automaticky vyplněn předpokládaný název souboru i jeho umístění. Je však třeba vybrat existující soubor, který chceme importovat.

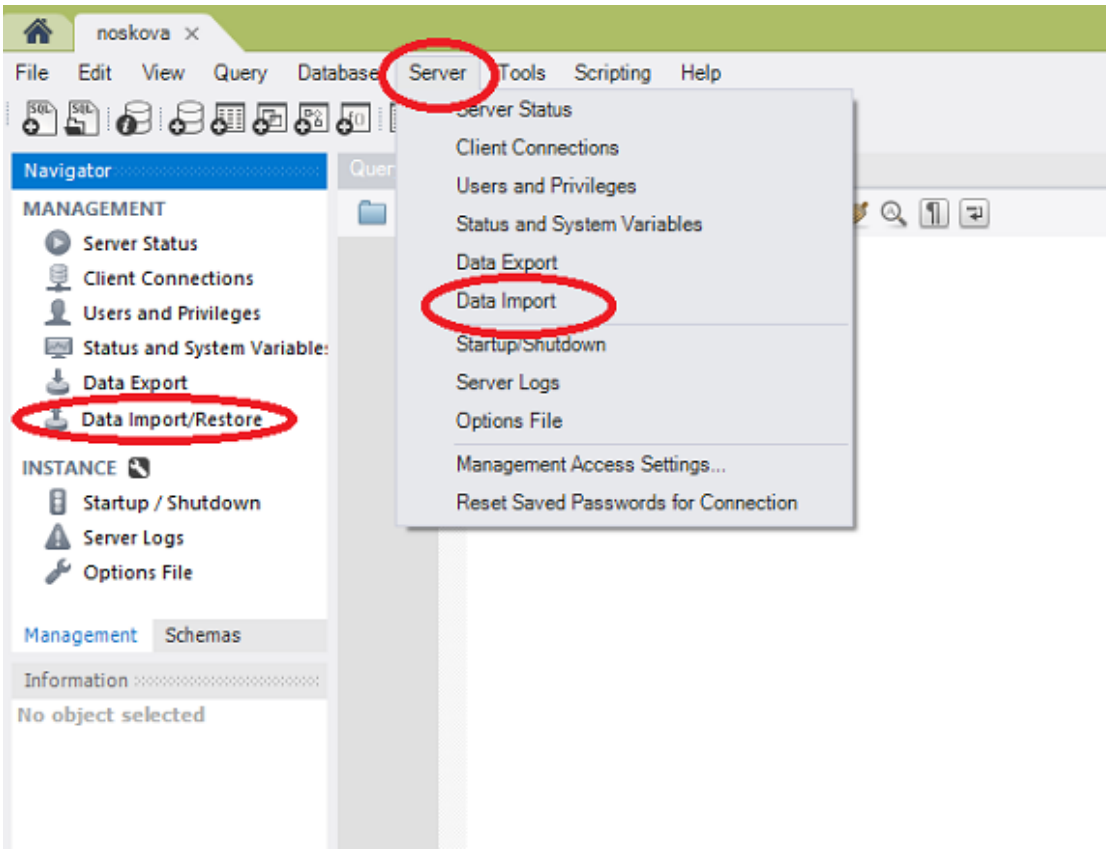
4) **Problém:** Zvolíme *Import from Self-Contained File* a spustíme import. Ten však skončí chybou.

[5] Soubor obsahuje databázi, jestliže obsahuje příkaz pro její vytvoření, tedy:

```
CREATE DATABASE název;
```

či

```
CREATE DATABASE IF NOT EXISTS název;
```



Obrázek 4: Import databáze ze souboru

Řešení: Je možné, že importovaný soubor neobsahuje databázi, ale pouze databázové objekty. V takovém případě musíme zvolit schéma, do kterého se bude importovat.

5) **Problém:** Po spuštění importu aplikace opakovaně vyžaduje heslo a do logů se vypisují chyby.

   Řešení: Tento problém pravděpodobně souvisí s uživatelskými právy. Kontaktujte administrátora databáze.

6) **Problém:** Aplikace nedovolí vytvořit nové schéma.

   Řešení: Tento problém pravděpodobně souvisí s uživatelskými právy. Kontaktujte administrátora databáze.

4.3.2 Import z jiného databázového systému

Databázi, tabulky i data lze importovat i z jiného databázového systému, např. PostgreSQL, Sybase ASE či MS SQL Server.

V MySQL Workbench zvolíme v hlavním menu *Database* a vybereme položku *Migration Wizard...* . Otevře se záložka pro migraci dat.

Před započetím procesu je nutné mít nainstalovaný příslušný ODBC ovladač[6]. Dále je potřeba mít nastavena práva pro migraci dat.

Postup migrace je jednoduchý. Požadované položky jsou jasně popsány, jednotlivé kroky navíc obsahují pomocné komentáře i bohatou nápovědu. Obecně je nejprve třeba se připojit ke zdrojovému i cílovému serveru. Ze zdroje pak vybereme databáze, které chceme migrovat. Nástroj analyzuje vybrané položky a nabídne uživateli seznam obsažených databázových objektů. Není nutné migrovat celou databázi, je možné zvolit pouze její část.

Pak dojde k převedení vybraných objektů do podoby, která je kompatibilní se systémem MySQL. Objeví-li se při převodu problém, nástroj MySQL Workbench nabídne možnost manuálně upravit vygenerovaný SQL kód před tím, než dojde k samotné migraci. Vygenerovaný, případně upravený kód můžeme buď importovat přímo do databázového systému anebo uložit do souboru.

Následuje samotný proces migrace. Vyskytne-li se problém, je k dispozici seznam chyb i varování. Podle toho je možné upravit problémové příkazy SQL manuálně a migraci dokončit. Proces je zakončen výpisem podrobného reportu.

4.4 Vykonání příkazů dotazovacího jazyka (SQL)

Fyzický datový model lze vytvořit nejen prostřednictvím grafického uživatelského rozhraní, ale i pomocí příkazů strukturovaného dotazovacího jazyka (SQL). Vyžaduje tedy alespoň minimální znalost tohoto jazyka. Čtenář se může seznámit se základy SQL či si připomenout jeho základní konstrukce v kapitole 2.5.

Po připojení k serveru se otevře instance, která mimo jiné obsahuje i záložku s vizuálním SQL editorem (defaultní název záložky je *Query 1*). Těchto záložek je možné otevřít víc a do každé psát část kódu či do některé psát kód a v jiné otevřít soubor. Součástí SQL editoru je nástrojová lišta (viz obr. 5).

[6] Rozhraní, přes které aplikace přistupují k datům v SŘBD. Pro jednotlivé databázové systémy se tento ovladač může lišit.

Co se týče tlačítek, která jsou umístěna v panelu nástrojů, je možné místo nich použít příkazy, které se skrývají ve volbách *Query* či *Edit* v hlavním menu. Tyto volby nabízejí i pár funkcí navíc, například převedení klíčových slov na malá či velká písmena.

MySQL Workbench umožňuje buď psát příkazy rovnou do editoru nebo je nahrát ze souboru či snippetu[7]. Chceme-li kód nahrát, v panelu nástrojů zvolíme tlačítko  a vybereme příslušný skript. V SQL editoru se pak zobrazí příkazy obsažené v souboru.

Nemáme-li soubor, který bychom nahráli, musíme psát SQL příkazy rovnou. Je dobré dbát na přehlednost kódu a dodržovat nějakou strukturu. Nástroj MySQL Workbench umožňuje nastavit barevné rozlišování jednotlivých částí kódu (například proměnných a klíčových slov).

Formát textu pak upravuje tlačítko  , které kód vhodně doplní o bílé znaky (mezery apod.) a zvýší tak jeho přehlednost. Velmi důležité je oddělovat jednotlivé příkazy středníkem. Jestliže nevíme přesnou posloupnost či složení jednotlivých příkazů,  žeme využít kontextovou nápovědu (oddíl *SQL Additions*, záložka *Context Help*), kde vybereme konkrétní funkci. Nápověda pak nabídne správnou syntaxi. Příkazy z editoru je také možné uložit do souboru. Tím je zajištěna možnost využít kód i příště. Pro uložení do souboru použijeme tlačítko s ikonou diskety.

Máme-li hodně obsáhlý kód, ve kterém je těžké něco najít, prostřednictvím tlačítka se zobrazí panel, který nám hledání usnadní.

Po napsání příkazů je nutné je vykonat. Tlačítko  způsobí vykonání všech příkazů v dané záložce SQL editoru. Tlačítkem  naopak dojde k vykonání pouze těch, které jsou umístěny za kurzorem. Dojde-li během vykonávání příkazů k chybě či proces nedoběhne do konce, je zobrazeno dialogové okno, které uživatele informuje, co se stalo a z jakého důvodu.

Panel nástrojů dále obsahuje tlačítka pro potvrzení (`COMMIT`) či odvolání (`ROLLBACK`) transakcí. Některé příkazy v sobě automatické potvrzení již obsahují. Jedná se o příkazy, které něco vytváří (`CREATE`), mění (`ALTER`) či mažou (`DROP`). Více

[7] Snippet je označení pro malý znovupoužitelný kus kódu. Najdeme je v záložce *Snippets* v oddílu *SQL Additions*. Postup pro vytváření snippetů viz kapitola 3.4.1.

o transakcích viz kapitola 2.1. Vykonáním příslušných příkazů dojde k vytvoření fyzického datového modelu.

4.4.1 Možné problémy a jejich řešení

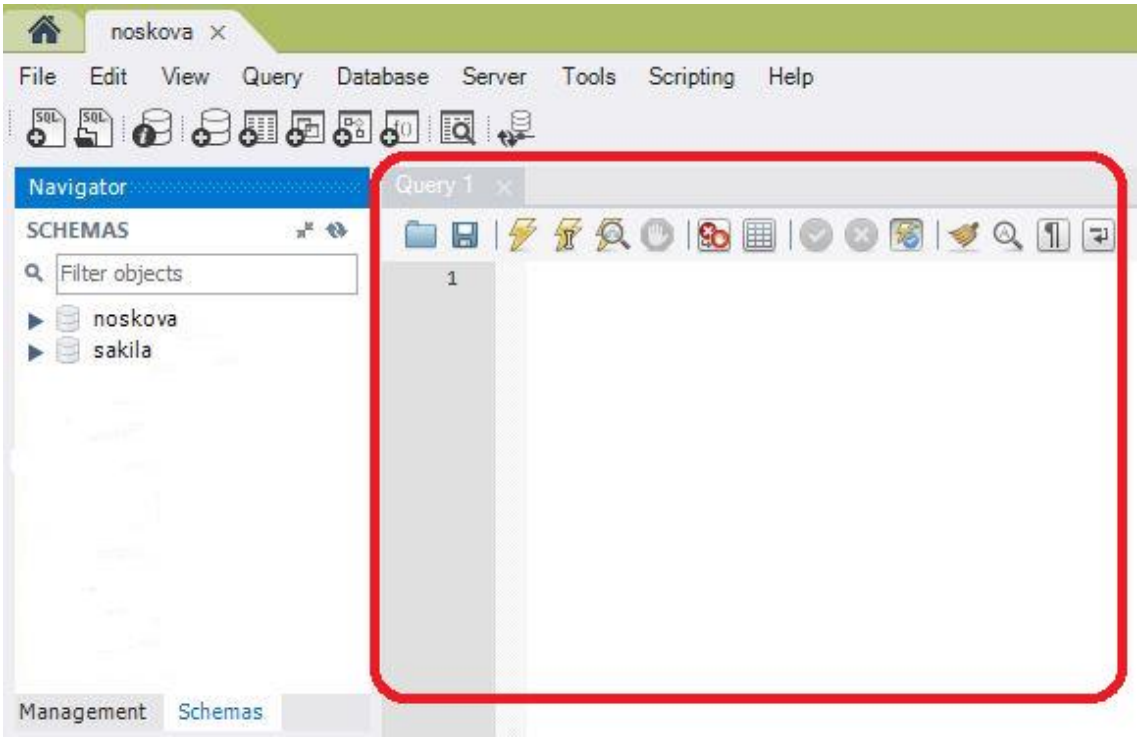
V případě vytváření databáze psaním SQL příkazů je nutné sledovat oddíl *Output* pod editorem. Po vykonání příkazů je zde vždy vypsána informace o úspěšnosti procesu, v případě výskytu chyby či varování pak jejich popis.

1) **Problém:** Některé příkazy či jejich části jsou v SQL editoru červeně podtržené.

   Řešení: Je třeba zkontrolovat syntaxi příkazů, někde se v nich vyskytuje chyba. Může se jednat i o vynechanou čárku či středník.

2) **Problém:** Zvolili jsme tlačítko s ikonou blesku pro vykonání příkazů, přesto ale nedošlo k vytvoření schématu.

   Řešení: V oddíle *Output* pod SQL editorem najdeme podrobnější informace. Příčina může být různá, například nedostatečná práva. V takovém případě je nutné kontaktovat administrátora. Problém ale také může nastat, jestliže se snažíme vytvořit objekt, který již existuje.



Obrázek 5: SQL editor

5 Zhodnocení nalezených cest

Tato kapitola obsahuje zhodnocení nalezených způsobů tvorby databáze. Hodnocení bylo provedeno analýzou výsledků dotazníku, který vyplnili čtyři dobrovolníci. Těm byl nejprve předán popis jednotlivých způsobů tvorby databáze (kapitola 4), obrázek ukázkového modelu s vysvětlivkami (Příloha A), manuál k nástroji MySQL Workbench (kapitola 3) a dotazník (Příloha C).

Úkolem dobrovolníků bylo nainstalovat MySQL Workbench (případně i systém MySQL), prostudovat dodaný materiál a na jeho základě vytvořit ukázkovou databázi všemi popsanými cestami. Na závěr pak jednotliví respondenti vyplnili dotazník. Na tento úkol měli dobrovolníci jeden kalendářní týden.

Po předání potřebných dokumentů bylo třeba se s některými respondenty osobně sejít a podrobněji vysvětlit účel tvorby a princip databází. Také bylo nutné pomoci s instalací nástroje MySQL Workbench a vytvořením uživatelských účtů i instancí pro připojení k MySQL serveru.

5.1 Sběr dat

Cílem analýzy a tvorby hodnocení je usnadnit čtenáři výběr nejvhodnějšího způsobu tvorby databáze v závislosti na jeho znalostech či zkušenostech. Podkladem pro tvorbu analýzy byly názory ostatních uživatelů nástroje MySQL Workbench, získané prostřednictvím dotazníku (viz Příloha D).

První část dotazníku se zaměřuje na získání základního přehledu o znalostech respondenta. V této části jsou proto otázky týkající se vzdělání, oboru zaměstnání či předchozích zkušeností s databázemi. Zbytek otázek se týká již samotného výzkumu. Respondenti hodnotí jednotlivé způsoby tvorby databáze, případně popisují jejich průběh. Předpokládá se, že dílčí hodnocení budou závislá na znalostech respondentů uvedených v první části dotazníku.

Dotazník byl vytvořen pomocí online nástroje na adrese *www.survio.cz*. Tento nástroj slouží pro tvorbu dotazníků i sběr a analýzu odpovědí. Respondenti obdrželi dotazník v elektronické podobě prostřednictvím e-mailu.

5.2 Řešený model

Úkolem respondentů bylo vytvořit databázi z dodaného návrhu modelu. Tento model subjekty použily pro všechny způsoby tvorby databáze.

Řešený model představuje zjednodušený systém tréninků jednotlivých sportů nějakého sportovního centra. Klienti mají možnost navštěvovat tréninky různých sportů. Jednotlivé sporty jsou vedeny jedním či několika trenéry s příslušnou kvalifikací. Tréninky probíhají v různých objektech, které obsahují jeden či více různých prostorů / místností. Například plavání se koná v bazénu, kondiční posilování v posilovně, volejbal v tělocvičně apod. Tréninky zároveň probíhají od pondělí do pátku v různých časech.

Model se skládá ze sedmi tabulek. Obrázek navrženého modelu lze najít v Příloze A.

V tabulce `Budova` najdeme názvy a adresy všech objektů, ve kterých se nachází nějaký prostor pro trénink. Může se jednat o bazény, horolezecké stěny, posilovny apod. V jedné budově, tedy na stejné adrese, se může nacházet několik využitelných prostorů.

Tabulka `Mistnost` uchovává informace o jednotlivých prostorách. Sleduje především jejich označení, dle kterého lze poznat typ objektu (například bazény mají na začátku písmeno ‘B’, tělocvičny ‘T’ apod.) a kapacitu. Dále pak obsahuje referenci na objekt (adresu), kde se nachází. V každé místnosti může probíhat několik různých sportů. Například k bazénu se mohou vztahovat tréninky jak kondičního tak synchronizovaného plavání. Sporty mohou probíhat na různých místech. Například na plavání lze chodit jak do bazénu A, tak do bazénu B, kdy oba tyto bazény nemusí být na stejné adrese. Místo, kde se trénink koná, zároveň ovlivňuje množství klientů, kteří se mohou na akci zapsat, tedy kapacitu.

Tabulka `Sport` obsahuje seznam sportů, které jsou nabízeny a na které se klienti mohou zapsat.

Tabulka `Trener` uchovává seznam zaměstnanců, kteří vedou tréninky nabízených sportů. Jsou sledovány údaje typu jméno, příjmení, bydliště apod. Jednotliví trenéři se specializují na konkrétní sporty, které také vyučují. Každý trenér vede různé tréninky na různých místech v různou dobu. Trenér nemůže vést dva tréninky ve stejnou dobu.

Tabulka `Klient` obsahuje seznam klientů sportovního centra. Zachycuje základní informace o osobách, jako jsou například jméno, příjmení, rodné číslo, bydliště či klientské číslo.

Tabulka `Trenink` obsahuje údaje o jednotlivých tréncích. Každá akce se může týkat pouze jednoho z nabízených sportů a koná se po určitou časovou jednotku v jedné z místností. Tréninky jednoho druhu sportu mohou probíhat současně, ale v různých prostorách a s různými trenéry. V každé místnosti se mohou konat tréninky různých sportů, nesmí se však časově překrývat. Protože čas a místo tréninků většinou zůstávají dlouhou dobu (v řádu měsíců) nezměněny, je nutné rozlišit, který trénink je a který není platný pro daný okamžik. Tuto informaci uchovává atribut `Platnost`.

Poslední tabulkou v řešeném modelu je `Ucastnici_treninku`. Ta udává, kteří klienti se zúčastní jakých tréninků. Protože rozvrhy zůstávají stejné či podobné i po několik měsíců a klienti většinou navštěvují stejné tréninky, obsahuje tabulka `Ucastnici_treninku` také časový údaj (rok, pololetí) zachycující účast klientů na jednotlivých tréncích. Klient může být zapsán na jednom tréninku max. jednou.

5.3 Analýza – znalosti respondentů

Pro provedení analýzy bylo osloveno několik respondentů různých znalostí. Průzkumu se zúčastnili pouze čtyři z nich. Respondenti se pohybují v těchto oborech: zdravotnictví, IT, elektro – sklad. Protože se z větší části jedná o lidi ve věku do 30 let, většina z nich ještě studuje, zároveň však již absolvovali nějakou praxi ve svém oboru.

5.3.1 Rozbor otázek

Jak hodnotíte úroveň svých dovedností s informačními technologiemi?

Odpovědi respondentů jsou dle očekávání. Subjekty pohybující se v oboru IT zvolili možnost „Jsou mi známy hlubší souvislosti (programátor, administrátor)“, zatímco subjekty z jiných oborů zvolili možnost „Informačním technologiím se raději vyhýbám.“.

S jakými technologiemi pracujete?

Odpovědi na tuto otázku ukazují, že v dnešní době se lze s informačními technologiemi setkat opravdu téměř všude. Každý z respondentů se setkává minimálně

s kancelářskými programy (např. MS Office), případně s informačním systémem v zaměstnání. Někteří navíc používají i vývojová prostředí.

Použil(a) jste někdy program pro tvorbu databází/databázový nástroj? Pokud ano, jaký?

Subjekty z IT oboru se s databázovým nástrojem již setkali, jak bylo předpokládáno. Používali například SQL Developer či MS SQL management studio. Zbytek respondentů zatím žádný nepoužil – nepočítaje toto testování v MySQL Workbench.

Jak hodnotíte své znalosti databází?, Jak hodnotíte svou znalost jazyka SQL?

Jak ukazují výsledky těchto otázek, pojmy „databáze“ a „SQL“ mohou být známy i lidem, kteří s nimi přímo nepracují. Hlubší znalosti však mají ti, kteří se pohybují v oboru IT.

5.3.2 Shrnutí

Označme respondenty jako subjekty A, B, C a D. Subjekt A se pohybuje v oboru IT. Databáze již vytvářel, dokáže vytvořit i složitější konstrukce jazyka SQL. S nástrojem MySQL Workbench se však ještě nesetkal. V oboru IT se pohybuje i subjekt B, který má však o databázích pouze základní povědomí a některé pojmy jsou mu cizí. Subjekt C se s databázemi setkal pouze nepřímo – prostřednictvím firemního informačního systému. Přesto však má o databázích základní povědomí. Subjekt D se také s databázemi setkal pouze prostřednictvím firemního informačního systému, o databázích však neví téměř nic.

5.4 Analýza – práce s MySQL Workbench

Druhá část dotazníku obsahuje zhodnocení od respondentů, týkající se procesů tvorby databáze popsanými způsoby. Předpoklad byl takový, že subjekty A a B dosáhnou lepších, možná i rychlejších, výsledků. Nebyly očekávány výrazné potíže respondentů, neboť v dokumentu jsou popsány základní a časté problémy i s jejich řešením.

5.4.1 Rozbor otázek

Kterými způsoby jste zkoušel(a) databázi vytvořit?

Úkolem respondentů bylo vytvořit databázi všemi uvedenými cestami. Počítala jsem však s možností, že některý ze způsobů bude pro respondenta příliš složitý či nepochopitelný, a tak se mu raději vyhne. Každý ze subjektů však nakonec vyzkoušel všechny popsané způsoby.

Pokud se Vám nepodařilo vytvořit databázi, napište, kterým způsobem jste to zkoušel(a) a v čem byl problém:

Odpovědi na tuto otázku představují jakési slovní shrnutí či hodnocení procesu tvorby databází vyzkoušenými postupy. Jak se ukázalo, v průběhu tvorby se u respondentů sice vyskytlo několik problémů, nakonec se jim však databázi podařilo vytvořit. Jako problémy zde subjekty uvádějí neporozumění textu, chybně nastavená práva či uživatelskou nepřívětivost prostředí.

Navrhoval(a)-li jste databázi pomocí logického datového modelu, kterým způsobem jste tento model vytvořil(a)?

Většina subjektů zvolila tvorbu databázových objektů nikoli prostřednictvím formulářů, ale umisťováním objektů v poli pro diagram. Tato metoda je dle mého názoru přehlednější hlavně tím, že uživatel vidí propojení jednotlivých tabulek díky relacím.

Který z vyzkoušených způsobů hodnotíte jako:

a) **Nejrychlejší** – každý respondent označil jako nejrychlejší jiný způsob tvorby databáze. Nelze tedy doporučit konkrétní cestu. Záleží na individuálním uživateli.

b) **Nejpomalejší** – jako nejpomalejší způsob byl větší částí respondentů označen způsob tvorby databáze bez návrhu logického datového modelu.

c) **Nejsložitější** – každý respondent označil jako nejsložitější jiný způsob tvorby databáze. Ani v tomto případě nelze vyvodit, který ze způsobů je nejsložitější.

d) **Nejjednodušší** – nejjednoduššími způsoby jsou dle respondentů import souboru a psaní SQL příkazů do editoru. Tato odpověď mě trochu zarazila, protože pro psaní příkazů je nutná znalost konstrukcí jazyka SQL, který většina respondentů

neovládá. Jak jsem se později dozvěděla, subjekty pro tuto metodu zvolili nahrání obsahu souboru, který byl určen k importu, do SQL editoru. Pak jen spustili vykonání příkazů.

Zaznamenal(a) jste rozdíl mezi vytvořenými databázemi? Pokud ano, uveďte jaký:

Vytvořené databáze by měly být stejné. Do dotazníku byla vložena tato otázka pro případ, že jsem něco přehlédla, neboť každý z respondentů se na databáze dívá jiným pohledem. Respondenti však jednohlasně uvedli, že na žádný rozdíl mezi vytvořenými databázemi nepřišli.

Jak hodnotíte nástroj MySQL Workbench?

Ukázalo se, že pro většinu respondentů je tento nástroj složitý, nepřehledný a uživatelsky nepřívětivý. Pouze subjekt A, který má širší povědomí o databázích, jej označil jako intuitivní.

5.4.2 Shrnutí

Nelze jednoznačně určit, který ze způsobů tvorby databáze je dle uživatelů nejrychlejší, nejpomalejší, nejjednodušší či nejsložitější. Na základě hodnocení jednotlivých respondentů lze vytvořit pouze doporučení. Máme-li k dispozici soubor s příkazy jazyka SQL pro tvorbu databáze, je doporučeno použít jej, ať už k importu či k nahrání a následné úpravě v SQL editoru. Obecně se nedoporučuje vytvářet databázi bez návrhu logického datového modelu.

6 Závěr

Cílem práce bylo nalézt různé způsoby tvorby databáze nástrojem MySQL Workbench. V úvodu je proto čtenář seznámen se základy databázového systému MySQL, který tvoří základ této práce. Dále má čtenář k dispozici manuál k modelování dat pomocí nástroje MySQL Workbench.

Nalezla jsem čtyři cesty vedoucí ke tvorbě databáze. Jedná se o její generování z logického datového modelu, tvorbu databáze pomocí formulářů a funkcí nástroje MySQL Workbench, import ze souboru či tvorbu kódu v SQL editoru. Jednotlivé cesty jsou popsány krok za krokem v kapitole 4. Každý ze způsobů zároveň obsahuje seznam několika možných problémů i jejich řešení.

Dalším cílem této práce bylo zhodnocení nalezených cest a doporučení nejvhodnější z nich. Hodnocení bylo provedeno pomocí čtyř respondentů různých znalostí, kteří dobrovolně otestovali a zhodnotili nalezené cesty.

Jednotliví respondenti označili jako nejjednodušší a nejrychlejší cesty import souboru a psaní příkazů do SQL editoru. Ty jsou však vhodné pouze v případě, máme-li k dispozici kód či skript, který bychom nahráli. Jako nejsložitější byla označena tvorba databáze bez návrhu logického datového modelu. Je to zřejmě tím, že uživatel nevidí grafické propojení jednotlivých tabulek modelu. Je doporučeno vytvářet databáze jejich vygenerováním z logického datového modelu.

Co se týče nástroje MySQL Workbench, většina respondentů jej označila jako složitý, nepřehledný a uživatelsky nepřívětivý. Nedoporučuji jej proto uživatelům, kteří jsou začátečníky v oblasti databází.

Literatura

[1] DUBOIS, Paul. c2003. *MySQL profesionálně: kompletní průvodce použitím, programováním a správou MySQL*. Praha: Mobil Media, 1071 s. IDnes internet knihy. ISBN 80-865-9341-X.

[2] GILFILLAN, Ian. 2003. *Myslíme v MySQL 4*. 1. vyd. Praha: Grada, 750 s. Knihovna programátora (Grada). ISBN 80-247-0661-X.

[3] *MySQL :: MySQL Editions*. (1. červenec 2015). Načteno z MySQL: http://www.mysql.com/products/

[4] *MySQL :: MySQL Workbench*. (1. červenec 2015). Načteno z MySQL: https://dev.mysql.com/doc/workbench/en/index.html

[5] WELLING, Luke a Laura THOMSON. 2005. *MySQL: průvodce základy databázového systému*. Vyd. 1. Brno: CP Books, 255 s. ISBN 80-251-0671-3.

Přílohy

Příloha A Řešený model

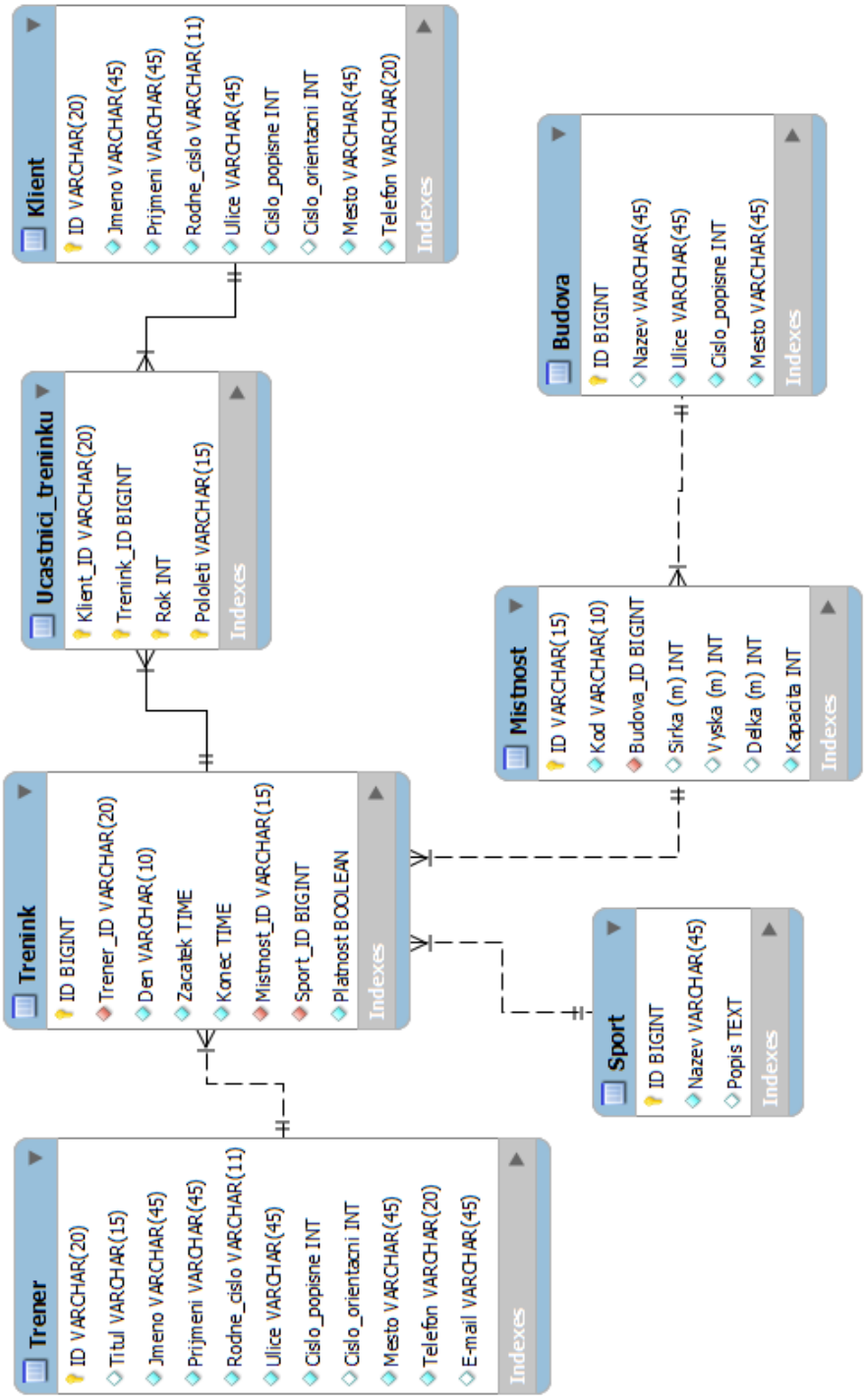
V ukázkovém ERA modelu jsou použity tyto ikony:

- Zlatý klíč - atribut je součástí primárního klíče.
- Plný kosočtverec - každý záznam musí mít u tohoto atributu vyplněnu nějakou hodnotu, tedy sloupec má nastavenou vlastnost `NOT NULL`.
- Prázdný kosočtverec - záznam nemusí mít u tohoto atributu vyplněnou hodnotu (není `NOT NULL`).
- Červený kosočtverec - atribut je součástí cizího klíče.
- Modrý kosočtverec - atribut není součástí cizího klíče.

Relace jsou označeny takto:

- Plná čára – relace typu *Identifying*
- Přerušovaná čára – relace typu *Non-Identifying*

Zakončení jednotlivých relací označuje kardinalitu vazby. V ukázkovém modelu jsou všechny vazby typu 1:N.



Obrázek 6: Diagram řešeného modelu

Příloha B Instalace a základní konfigurace

Systém MySQL včetně nástroje MySQL Workbench lze zdarma stáhnout na adrese *http://dev.mysql.com/downloads/*. Pro stažení není nutné mít u společnosti založený účet. K dispozici jsou vždy pouze nejnovější verze, starší verze je možné nalézt v archivu.

Při instalaci je třeba zvolit heslo pro připojení k vytvořenému MySQL serveru. Je nezbytné si toto heslo pamatovat!

MySQL server

Označme MySQL server jako počítač, který slouží jakožto úložiště databází či dat. Tento server je však současně i programem, který na daném počítači běží a poskytuje klientům specifické služby související s databázovým systémem MySQL. Uživatelé, kteří chtějí přistupovat k datům na serveru, se k němu připojují prostřednictvím klienta. Existuje několik možností nastavení:

1) klient a server se nachází na různých strojích,
2) klient i server se nachází na stejném stroji.

V této práci se předpokládá druhá ze zmíněných možností, tedy klient i server na stejném počítači.

Pokud to při instalaci zvolíme, bude MySQL server při prvním spuštění běžet. Pokud ne, je třeba jej spustit.

Spuštění MySQL serveru

Ve složce *Ovládací panely* najdeme položku *Nástroje pro správu* a v ní potom položku *Služby*. Zde vybereme MySQL server a spustíme jej. Výchozí název serveru je MySQL56, pokud při instalaci nezvolíme jinak.

Další možností, jak spustit server, nabízí samotný program MySQL Workbench. Na úvodní stránce zvolíme v oddílu *MySQL Connections* konkrétní instanci a zadáme heslo. Na nově otevřené stránce se v panelu *Navigator* přepneme do záložky *Management*. Zvolíme možnost *Startup/Shutdown* a otevře se nová záložka. Server spustíme tlačítkem *Start Server*.

Konfigurace

Při prvním spuštění je dobré nakonfigurovat celou aplikaci. Lze to však provést kdykoli v průběhu běhu programu. Pro konfiguraci slouží položka *Edit* v hlavním menu a volba *Preferences*. Lze nastavit například font a barvu písma pro SQL příkazy či výchozí názvy vytvořených tabulek.

Tvorba uživatelských účtů

Administrátor musí vytvořit účet pro každého uživatele, který bude s databází pracovat. Na úvodní stránce v oddíle *MySQL Connections* se administrátor přihlásí k serveru prostřednictvím výchozí instance. Je otevřena nová záložka. V panelu *Navigator* je třeba zvolit *Users and Privileges,* případně zadat heslo, vyžaduje-li to aplikace. Otevřená záložka obsahuje jednak seznam již vytvořených uživatelských účtů, ale také prostor pro definici nového účtu. Pomocí tlačítka *Add Account* přidáme nový uživatelský účet. V záložce *Login* pak definujeme přihlašovací jméno uživatele (*Login Name)* a heslo (*Password*).
V záložce *Schema Privileges* pak účtu přiřadíme schéma, čili prostor na serveru pro tvorbu databází. Klikneme na *Add Entry*… a zvolíme jedno či více schémat.
Dalším krokem je přiřazení práv ke zvolenému schématu. Ty zvolíme na základě role uživatele. Vše dokončíme kliknutím na tlačítko *Apply*.

Nakonec je třeba vytvořit uživateli instanci pro připojení k databázi. Na úvodní obrazovce klikneme na ikonu malého plus v oddílu *MySQL Connections*. Zobrazí se okno, ve kterém založíme nové připojení. Zvolíme název (*Connection Name* - například uživatelské jméno uživatele), uživatelský účet (*Username*) a heslo (prostřednictvím tlačítka *Store in Vault…).* Dále je nutné doplnit údaje pro spojení se serverem (IP adresu a port). Ke každé vytvářené instanci můžeme přiřadit schéma. Proces tvorby připojení dokončíme tlačítkem *OK*.

Příloha C Dotazník