
**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**
**BRNO UNIVERSITY OF TECHNOLOGY**

**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**
**FACULTY OF INFORMATION TECHNOLOGY**

**ÚSTAV INTELIGENTNÍCH SYSTÉMŮ**
**DEPARTMENT OF INTELLIGENT SYSTEMS**


# DATABÁZE PRO GENEALOGICKÉ MODELY

**DATABASE FOR GENEALOGICAL MODELS**

**DIPLOMOVÁ PRÁCE**
**MASTER'S THESIS**


**AUTOR PRÁCE** **Bc. JAKUB MAJZLÍK**
**AUTHOR**

**VEDOUCÍ PRÁCE** **Ing. RADEK KOČÍ, Ph.D.**
**SUPERVISOR**



**BRNO 2022**

**Vysoké učení technické v Brně**
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2021/2022

# Zadání diplomové práce


Student: **Majzlík Jakub, Bc.**
Program: Informační technologie a umělá inteligence
Specializace: Informační systémy a databáze
Název: **Databáze pro genealogické modely**
**Database for Genealogical Models**
Kategorie: Databáze
Zadání:

1. Seznamte se s problematikou práce s genealogickými daty, tvorbou genealogických modelů, zejména rodokmenů, a existujícími databázemi v rámci projektu DEMoS.
2. Analyzujte stávající databáze s ohledem na potřeby projektu DEMoS, zejména velký objem dat, zachování původní podoby přepisovaných záznamů včetně chyb a nepřesností a důraz na rychlost a správnost vyhledávání záznamů. Zaměřte se také na vhodnost použitých databázových systémů.
3. V souladu s provedenou analýzou revidujte stávající databázové schéma.
4. Doplňte databázové schéma o dosud nezpracované genealogické zdroje (urbáře, lánové rejstříky apod.) a převeďte základní data o těchto zdrojích do databáze systému DEMoS.
5. Navrhněte a realizujte úpravu uživatelského rozhraní pro správu genealogických modelů včetně integrace do webového rozhraní systému DEMoS.
6. Vyhodnoťte realizované změny databáze s ohledem na předem stanovená kritéria (potřeby projektu) a proveďte uživatelské testování aplikace.

Literatura:

- Moravský Zemský Archiv. Digitalizace archivních fondů. http://www.mza.cz/a8web/a8apps1/a8apps1.htm.
- KOČÍ Radek, ROZMAN Jaroslav a ZBOŘIL František. Database Concept for Transcription of Registry Records into Digital Form. In: *Proceedings of the 3rd International Conference on Software Engineering and Information Management - ICSIM'20*. Sydney: Association for Computing Machinery, 2020, s. 21-25. ISBN 978-1-4503-7690-7.

Při obhajobě semestrální části projektu je požadováno:

- První tři body zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/
Vedoucí práce: **Kočí Radek, Ing., Ph.D.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 18. května 2022
Datum schválení: 3. listopadu 2021

# Abstrakt

Systém DEMoS je genealogická databáza, ktorá slúži na prepis záznamov z rôznych historických prameňov. Momentálne podporuje len prepis z matrík. Prvým cieľom tejto práce je analyzovať databázový systém a schémy tabuliek využívané systémom DEMoS. Pozriem sa na vhodnosť zvolených dátových typov a na výskyt indexov, ktoré by malo pomôcť s výkonom databázy. Keďže si tento systém za cieľ stanovuje prepis aj z iných zdrojov ako sú matriky, tak budem pracovať na návrhu integrácie ďalších historických zdrojov. Integrovať sa budú zdroje „Urbáře“, „Berní rula“, „Lánové rejstříky“ a „Poddánska přiznávací fasé“. Bude potrebné navrhnúť nové tabuľky pre tieto zdroje a naimplementovať potrebné triedy do systému (napríklad triedy, ktoré komunikujú s databázou). Takisto bude potrebný zásah do užívateľského rozhrania a implementácia chýbajúcich pohľadov.



# Abstract

The DEMoS system is a genealogical database that is used to transcribe records from various historical sources. Currently, the system only supports transcripts from matrices. The first goal of this thesis is to analyze database system and table schemas used in system DEMoS. I will look at the suitability of the selected data types and the occurrence of indexes, which should help with the database performace. Since this system aims to transcribe from more sources then matrices, I will work on a proposal for the integration of other historical sources. The sources of „Urbář“, „Berní rula“, „Lánové rejstříky“ and „Poddánska priznávací fasé“ will be integrated. I will need to design new tables for these resources and implement necessary classes in the system (for example, classes that communicate with database). I will also make nessesary changes to the user inteface and implement missing views.



# Kľúčové slová

SQL, NoSQL, MySQL, genealógia, DEMoS, php, Nette

# Keywords

SQL, NoSQL, MySQL, genealogy, DEMoS, php, Nette


# Citácia

MAJZLÍK, Jakub. *Databáze pro genealogické modely.* Brno, 2022. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Radek Kočí, Ph.D.

# Databáze pro genealogické modely

## Prehlásenie

Prehlasujem, že som tuto diplomovú prácu vypracoval samostatne pod vedením pána Ing. Radka Kočiho Ph.D. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Jakub Majzlík
12. mája 2022


## Poďakovanie

Rád by som poďakoval vedúcemu práce Ing. Radekovi Kočimu, Ph.D. za odborné vedenie pri písaní tejto práce a za konzultácie.

# Obsah

# Kapitola 1

# Úvod

Genealógia je vedná disciplína, ktorá pracuje s historickými prameňmi. Týchto prameňov sa za tie všetky roky môže nazhromaždiť veľké množstvo a pre analýzu za pomoci počítačov je ich potrebné zdigitalizovať. Na tieto účely, presnejšie na zdigitralizovanie záznamov z matrík, slúži aj systém DEMoS. Tento systém má fungovať pre širokú verejnosť, teda dané prepisy vykonávajú užívatelia systému. Momentálne sa v databáze nachádza vyše 95620 záznamov a predpokladá sa, že v budúcnosti ich môže byť viac ako 125 miliónov. Aby bola práca s týmto množstvom dát rýchla, je potrebné dbať na vhodne vybraný databázový systém a na vhodne navrhnutú schému jednotlivých tabuliek. Navyše systém DEMoS si za cieľ stanovuje možnosť prepisu nielen z matrík, ale aj z iných historických prameňov. Akurát možnosť prepisu z iných zdrojov momentálne v systéme chýba.

Aj kvôli faktom spomenutých v odseku vyššie, je potrebná analýza databázového systému, čo je mojou prvou úlohou. V súčasnosti DEMoS používa relačný databázový systém MySQL, no pozrel som sa na ďalšie dva, a to Oracle Database a PostgreSQL. Mimo relačných je vhodné sa zamyslieť aj na možnosti použitia nerelačných databáz, ktoré by mohli svojimi vlastnosťami byť vhodnejšími kandidátmi na použitie.

V systéme sa nachádza veľké množstvo tabuliek s veľkým počtom atribútov. Pozriem sa na použité dátové typy a navrhne možné úpravy, ktoré by mohli viesť k zlepšeniu výkonnosti. Medzi ďalšie úpravy, na ktoré sa pozriem sú indexy a optimalizácia procedúry používanej na získanie kompletného záznamu o narodení človeka.

Spomínaná analýza databázových systémov a aktuálne používaných schém tabuliek, je len časť mojej práce. Hlavnou náplňou tejto práce je upraviť systém DEMoS tak, aby sme boli schopný ho použiť na prepis aj iných záznamov ako sú spomínané matriky. Ďalšími historickými prameňmi, z ktorých bude možný prepis do našej databázy sú „Urbáře“, „Berní rula“, „Lánové rejstříky“ a „Poddánska přiznávací fasé“. Bude potrebné navrhnúť nové tabuľky, ktoré sa budú požívať a následne vykonať zásah do systému. Bude potrebné naimplementovať chýbajúce pohľady, komponenty a triedy potrebné na komunikáciu s databázou. A na koniec bude potrebné všetky tieto zmeny otestovať.

Štruktúra tejto diplomovej práce bude pozostávať z kapitoly zaoberajúcej sa stručným popisom genealógie (kapitola 2) a ďalej sa tu bude nachádzať kapitola o samotnom systéme DEMoS (kapitola 3). Nasleduje kapitola, ktorá sa bude zaoberať analýzou databázových systémov (kapitol 4) a revízia aktuálnych schém spolu so spomínanou procedúrou (kapitola 5). Pokračovať budem s návrhom nových tabuliek potrebný pre ďalšie zdroje (kapitola 6) a samotná implementácia (kapitola 7). Toto všetko zakončím otestovaním novo vykonaných zmien (kapitola 8) a záverom tejto práce, kde zhrniem dosiahnuté ciele (kapitola 9).

# Kapitola 2

# Genealógia

Genealógia je pojem, ktorý označuje vednú disciplínu zaoberajúcu sa vzťahmi medzi ľuďmi, ktorí majú spoločný rodový pôvod. Táto veda patrí medzi takzvané pomocné historické vedy[27]. Počiatky tejto vednej disciplíny siahajú až do starovekého Egypta, kde sa na príklad viedli záznamy o faraónoch a ich rodinách[30]. Genealógovia zisťujú a určujú tieto vzťahy na základe rôznych historických prameňov. Medzi tie staršie historické pramene, z ktorých môžu genealógovia čerpať, patria rôzne zápisky, listiny a úradné knihy. K zdrojom bádania môžu slúžiť aj denníky alebo aj rodinné albumy. Záznamy od polovice 17. storočia môžeme nájsť v matrikách, ktoré boli zavedené na území Českej republiky reformou Jozefa II[40].

Medzi moderné spôsoby, akými je možné získať informácie o predkoch je genetická genealógia. Genetická genealógia využíva analýzu DNA k získavaniu napríklad informácií o tom, či dvaja ľudia mali rovnakého predka[1]. Existuje veľa spoločností, ktoré za poplatok ponúkajú získanie rôznych informácií o predkoch práve z DNA. Väčšinou po objednaní testu vám príde domov súprava, s ktorou si odoberiete vzorku DNA a odošlete ju späť firme, ktorá ju následne vyhodnotí. Jednu z týchto spoločností stručne popíšem v kapitole 2.2.1.

## 2.1 Genealogické modely

Genealogické modely sú grafické modely, ktoré znázorňujú jednotlivé osoby určitého rodu. Medzi tieto modely môžeme zaradiť napríklad rodokmeň, vývod alebo aj rozrod. Pri tvorbe týchto modelov sa môžeme stretnúť s osobou, ktorú označujeme ako *probant*. Probantom rozumieme osobu, ktorá je v danej hierarchii ako východisková (môže sa jednať napríklad o človeka, ktorý si dáva vypracovať rodokmeň)[31]. Takisto existujú rôzne zaužívané symboly pri tvorbe týchto modelov. Medzi tieto symboly patrí napríklad[1]:

- * – označuje narodenie jedinca
- + – označuje úmrtie jedinca
- ∞ – označuje svadbu dvoch jedincov
- □ - označuje mužské pohlavie
- ○ - označuje ženské pohlavie

#### Rodokmeň

Rodokmeň je najjednoduchší genealogický model, ktorý zobrazuje príbuzných určitej rodovej línie. Jednotliví potomkovia sú zoradení od probanta a to od minulosti do prítomnosti. V rodokmeni sa nachádzajú len potomkovia v mužských líniách, teda s rovnakými priezviskami. O ženách sa tu nachádza minimum informácií a nenachádzajú sa tu informácie o nemanželských a adoptovaných deťoch[30].

#### Vývod

Vývod funguje na podobnom princípe ako rodokmeň s výnimkou toho, že sa v ňom zaznamenávajú všetky línie, teda nielen mužské, ale aj ženské. Vývod je najpoužívanejším genealogickým modelom, ktorý sa používa a často sa zobrazuje vo forme stromu, kde probant sa nachádza v koreni a jeho predkovia sa postupne vetvia[2].

#### Rozrod

Rozrod je najkomplikovanejší genealogický model, z tohto dôvodu je veľmi náročné jeho spracovanie. Obsahuje všetkých priamych potomkov východiskového páru, čo môže zahrňovať až niekoľko tisíc osôb. K zložitosti prispieva aj to, že sa tu nachádzajú aj nemanželské deti a ich potomkov. Vedený je od najstaršej osoby rodu, až do súčastnosti[2].

## 2.2 Tvorba rodokmeňov

Základom tvorby rodokmeňu je mať informácie o svojich predkoch. Svoje bádanie môžeme začať už doma, kde si nazhromaždíme všetky dostupné rodinné pramene. Môže sa jednať o staré listy a dokumenty našich starých a prastarých rodičov, ktoré sme našli na povale alebo rôzne rodné a úmrtné listy. Je dobré začať tvoriť rodokmeň od seba a žijúcich príbuzných, veď predsa je ľahšie sa opýtať našej babičky, či nemá rodný alebo úmrtný list po svojej babičke, ako pátrať v archívoch. No hlavne problémom by bolo pristúpiť k záznamom v matrikách, ktoré sú mladšie ako 100 rokov. Tieto záznamy z dôvodu ochrany osobných údajov nie sú dostupné verejnosti. Avšak do záznamov starších ako 100 rokov už môžeme nahliadnuť a to buď osobne navštívením matriky alebo aj online, keďže väčšina matrík je už zdigitalizovaných.

Pre tvorbu rodokmeňu nám stačí len pero a papier, avšak toto nie je ideálna cesta. Predsa len existujú oveľa prehľadnejšie spôsoby tvorby s množstvom užitočných funkcií, ako napríklad použitie počítačového programu na to určenému. Existuje aj množstvo online nástrojov, ktoré nám môžu pri tvorbe rodokmeňu pomôcť, patria medzi ne napríklad MyHeritage[1], Ancestry[2] alebo aj online index FamilySearch[3].

### 2.2.1 MyHeritage

Medzi známe nástroje pre tvorbu rodokmeňov patria nástroje od spoločnosti MyHeritage. Spoločnosť MyHeritage vznikla už v roku 2003, založil ju Gilad Japhet vo svojom dome a svoje služby ponúka bezplatne už od roku 2005. Táto služba má k dispozícii viac ako 16 miliárd historických záznamov. Ak si chceme cez túto službu vytvoriť rodokmeň stačí

[1] www.myheritage.com
[2] www.ancestry.com
[3] www.familysearch.org

nám poznať meno, priezvisko, bydlisko a dátum narodenia a úmrtia nášho predka[4]. Takisto MyHeritage ponúka od roku 2016 zistenie informácii o predkoch prostredníctvom DNA testu[12]. K dispozícii je nielen online verzia, ale aj desktopová verzia na stiahnutie do počítača (MyHeritage Family Tree Builder). MyHeritage ponuka aj možnosť importovania a exportovania rodokmeňu vo formáte GEDCOM.

Ako som už spomínal, tak MyHeritage je k dispozícii zdarma, no v ponuke sú aj platené verzie nástroja, ktoré prinášajú rôzne výhody, ako napríklad možnosť tvorby väčšieho stromu (základná verzia umožňuje mať v rodokmeni len 250 osôb), prístup k miliónom historických záznamov, nástroje na kontrolu konzistencie stromu a iné[13].

### 2.2.2 FamilySearch

Ďalším zaujímavým nástrojom, ktorý môžeme použiť pri tvorbe rodokmeňa je stránka FamilySearch. Na tejto stránke je možne získať informácie o osobe, ako napríklad jeho rodičoch a súrodencoch, vyplnením niekoľkých základných údajov. Medzi tieto údaje patrí meno, priezvisko, rok narodenia a názov bydliska [5]. Informácie o osobe, prípadne snímky dokumentov spojených s hľadanou osobou, je možné bezplatne získať. Pre niektoré operácie je potrebné sa zaregistrovať.

[4]Platí pre platenú verziu.
[5]Je možné niektoré z týchto údajov vynechať.

# Kapitola 3

# Systém DEMoS

## 3.1 O systéme

DEMoS je genealogická databáza, ktorá bola vyvíjaná na našej fakulte informačných technológii v spolupráci s Ústavom pomocných vied historických a archívnictva Filozofickej fakulty Masarykovej univerzity v Brne. Tento systém slúži na prepis informácií z rôznych archívnych prameňov ako napríklad z matrík, pozemkových kníh a iných prameňov. Z prepísaných záznamov sa budú môcť potom vytvárať rôzne sociálne alebo genealogické modely, ktoré budú zahrňovať nielen príbuzné osoby, ale aj svedkov pri svadbe alebo osobu, ktorá krstila dieťa a iné[39].

K prepisovaniu môže prispieť hocikto, ako je udané na stránkach projektu, databáza projektu DEMoS je koncipovaná ako komunitná a k vytvoreniu nového záznamu je potrebné si len vytvoriť účet. Po pridaní záznamu, záznam neprechádza žiadnou kontrolou, no neskôr ho môžu skontrolovať a prípadne upraviť iný používateľ. Každý používateľ má svoju úroveň, ktorá sa pridávaním a úpravou záznamov postupne zväčšuje. Používateľ môže upraviť záznam, ak má vyššiu úroveň ako autor záznamu. Je zrejmé, že budú vznikať chyby, no môžu sa minimalizovať tým, že sa budú používať nejaké všeobecné zásady prepisu. Tieto pravidlá môžeme nájsť na stránkach projektu DEMoS [1][39].

Keďže DEMoS je webová aplikácia, využíva technológie vhodné pre web. Samotný systém je teda napísaný v programovacom jazyku PHP s využitím Nette Web Framework. DEMoS pre ukladanie záznamov využíva relačný databázový systém MySQL. Tento databázový systém bude bližšie popísaný v kapitole 4.1.1.

Aplikácia donedávna bežala na serveri VUT s názvom Perun. S týmto serverom boli problémy, tak sa rozhodlo o prechode na nový, výkonnejší server s názvom Radegast. Aplikácia teda aktuálne beží na adrese http://radegast.fit.vutbr.cz/

### PHP

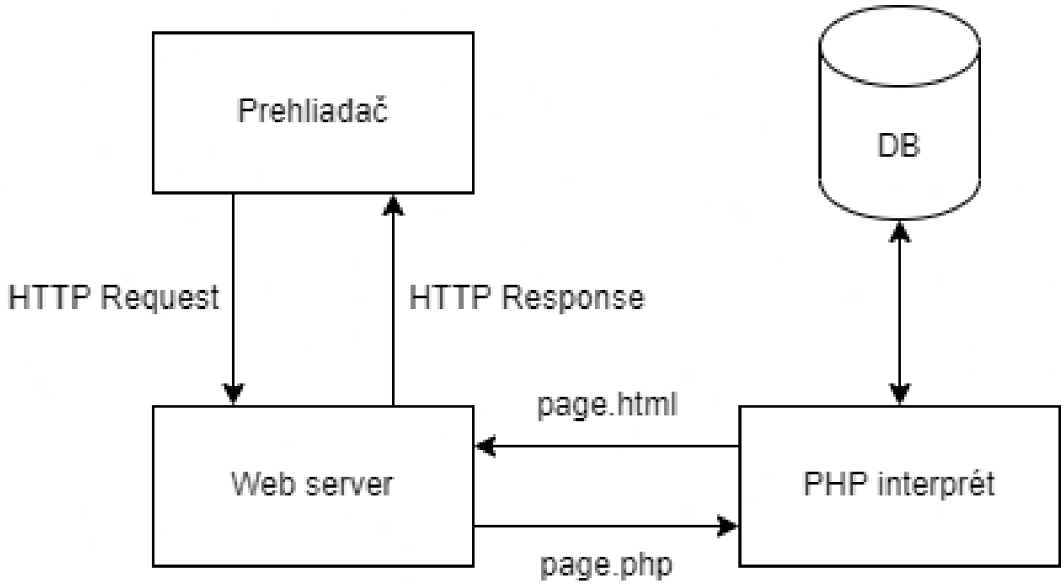
PHP (akronym pre PHP: Hypertext Preprocessor) je skriptovací jazyk, ktorý je hlavne používaný pri tvorbe webových aplikácií, presnejšie pre tvorbu skriptov, ktoré bežia na strane serveru. Avšak nie je to jeho jediné využitie. Tento jazyk má využitie aj pri tvorbe skriptov, ktoré majú bežať v príkazovom riadku a dokonca je možné v ňom tvoriť aj GUI aplikácie s využitím *PHP-GTK* knižnice[9]. Tento jazyk vznikol v roku 1994, je dodnes jeden z najpoužívanejších jazykov pre tvorbu programov bežiacich na strane serveru a je distri-

[1] http://radegast.fit.vutbr.cz/napoveda/pravidla-prepisu/

buovaný pod open-source licenciou *PHP License v3.01*[2]. Aktuálne najnovšia verzia, ktorá je k dispozícii na stiahnutie je verzia 8.1.5.

Skriptovací jazyk PHP je možné využívať vo všetkých najpoužívanejších operačných systémoch (Linux, Windows, macOS, ...) a vo väčšine webových serveroch (Apache, IIS, ...). PHP podporuje či procedurálne, tak aj objektovo-orientované programovanie. Výhodou je aj podpora veľkého množstva databázových systémov[7].

Na obrázku 3.1 môžeme vidieť životný cyklus požiadavku na webovú stránku, ktorá obsahuje PHP kód. Keď užívateľ chce pristúpiť na stránku, prehliadač odošle HTTP požiadavku na web server. Ak požadovaná stránka obsahuje PHP kód, web server si zavolá PHP interpret, ktorý vykoná dané inštrukcie. PHP interpret ďalej môže komunikovať s inými službami (napríklad s databázou) a interpretuje prijatý kód. Následne vygeneruje odpoveď vo forme HTML dokumentu[3], ktorú predá web serveru a ten ju odošle prehliadaču[28].




Obr. 3.1: Životný cyklus požiadavky na stránku s PHP kódom

### Nette Web Framework

Nette je skupina rôznych knižníc a frameworkov, ktoré programátorovi uľahčujú tvorbu webových aplikácií v programovacom jazyku PHP. Jedná sa o jeden z najpoužívanejších frameworkov používaných na tvorbu webových aplikácií v Českej republike, čomu aj prispieva fakt, že autor pochádza práve z tejto zeme. Môžeme ho použiť na tvorbu či jednoduchých webov určených na prezentáciu, tak aj zložitých e-shopov alebo CMS systémov. S tvorbou aplikácií v tomto frameworku je jednoduché začať. Autori si dali záležať na bohatej dokumentácií, ktorá obsahuje popisy použitých princípov, návody a postupy k jednotlivým nástrojom a modulom tohto frameworku, riešenia najčastejších problémov a iné. S tvorbou aplikácie môžeme jednoducho začas s použitím nástroja `Composer`, ktorý slúži na manažovanie knižníc v projekte.

[2] https://www.php.net/license/3_01.txt
[3] Odpoveď môže byť aj v inom formáte, napríklad XML alebo aj pdf

Použitie Nette má mnoho výhod:

- Množstvo zložitých problémov rieši Nette za nás, ako napríklad implementáciu autentifikácie, routovanie a iné
- Je moderný a podporuje HTML5, AJAX a SEO
- Obsahuje ochranu pred rôznymi útokmi (CSRF, XSS, Session hijacking a iné)
- K dispozícii sú pokročilé nástroje na testovanie a ladenie aplikácie
- Veľká komunita programátorov
- Prepracovaná dokumentácia nielen v anglickom jazyku, ale aj v českom jazyku

Tento framework je spravovaný organizáciou Nette Foundation a môžeme ho používať pod dvoma licenciami a to buď pod BSD licenciou alebo GPL licenciou[16].

Aplikácia, ktorá je napísaná v tomto frameworku, využíva architektúru MVP (Model-View-Presenter), čo je obdoba architektúry MVC (Model-View-Controller). Podstatou tohto návrhového vzoru je separácia užívateľského rozhrania a biznis logiky aplikácie, čo nám umožní jednoduchšie testovanie a vývoj aplikácie. Architektúra teda pozostáva z troch vrstiev[42]:

- **Model** – Vrstva programu, ktorá uchováva biznis logiku, dáta a stav aplikácie.
- **View** – Vrstva programu, ktorá sa stará o zobrazenie dát užívateľovi, požadované dáta získava od vrstvy *Presenter*.
- **Presenter** – Vrstva programu, ktorá reaguje na požiadavky zo strany *View* a s pomocou vrstvy *Model* ich obsluhuje. *Presenter* je takou medzivrstvou medzi pohľadom a modelom.

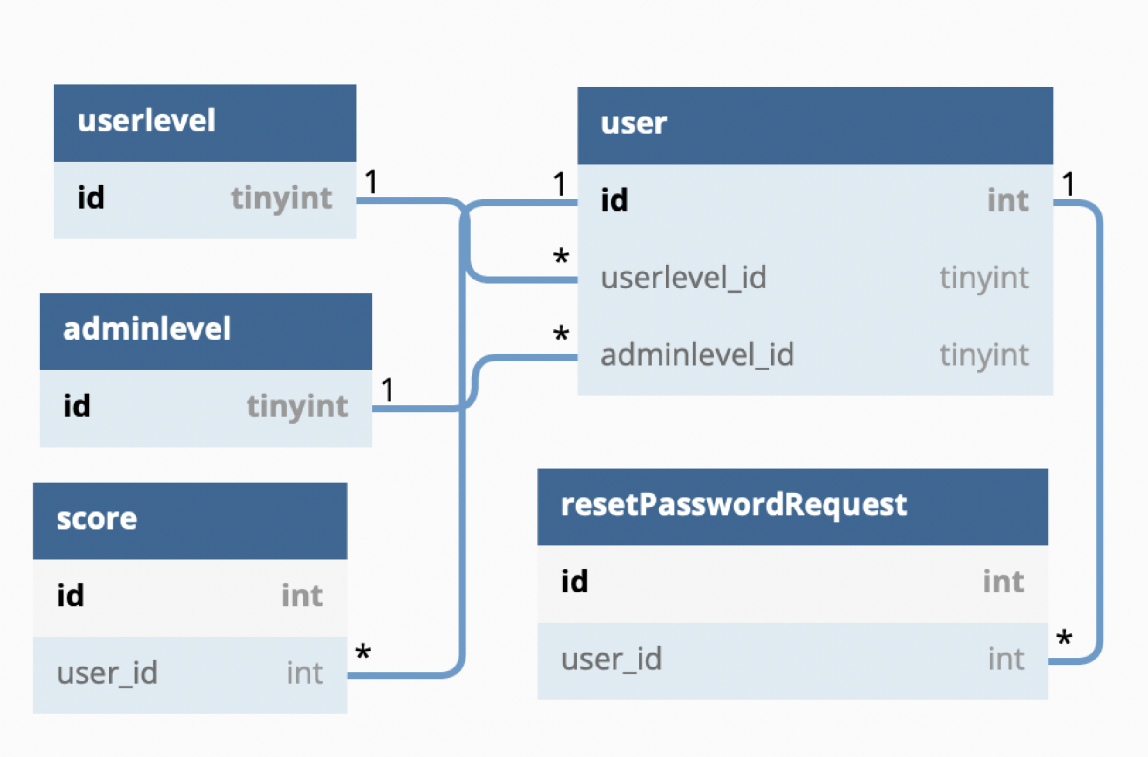
Nette obsahuje množstvo potencionálne užitočných nástrojov a systémov, ako napríklad Tracy. Tracy je nástroj určený na ladenie kódu, ktorý podľa autorov patrí k špičke medzi diagnostickými nástrojmi. Tento nástroj nám pomôže či už s odhalením a logovaním chýb, ale aj s meraním časovej náročnosti skriptov a databázových dotazov alebo sledovaním pamäťových nárokov aplikácie[18].

Ďalším zaujímavým Nette projektom je aj Latte, ktorý slúži ako šablónovací systém pre PHP. Obsahuje radu zaujímavých vlastností, ako napríklad ochrana pred XSS útokmi, kontrola odkazov, možnosť znovupoužitia šablón a iné. S pomocou Latte je možné generovať nielen HTML súbory, ale aj XML, CSV, iCal a iné[17].

## 3.2 Aktuálny stav databázy

Aktuálne sa v databáze 42 tabuliek, ktoré by sme mohli rozdeliť na tri typy. Prvým typom tabuliek sú tie, ktoré obsahujú údaje o používateľoch systému DEMoS. Zjednodušenú schému len s primárnym kľúčom a cudzími kľúčmi je možné vidieť na obrázku 3.2. Samotné tabuľky so stručným popisom je možné vidieť v nasledovnom zozname:

- **user** – tabuľka s údajmi o registrovanom používateľovi (meno, priezvisko, email, . . . )
- **resetPasswordRequest** – tabuľka s údajmi pre obnovenie hesla pre používateľa (token, čas expirácie a id užívateľa)
- **score** – tabuľka, ktorá obsahuje počet bodov, ktoré daný používateľ získal za pridávanie záznamov do databázy v jednotlivých jazykoch (lat, ge, cz, sk, pl a sc).
- **adminlevel** – tabuľka, kde sa nachádzajú rôzne úrovne pre administrátorov systému (názov a úroveň)
- **userlevel** – tabuľka, kde sa nachádzajú úrovne pre používateľov systému (názov a úroveň)

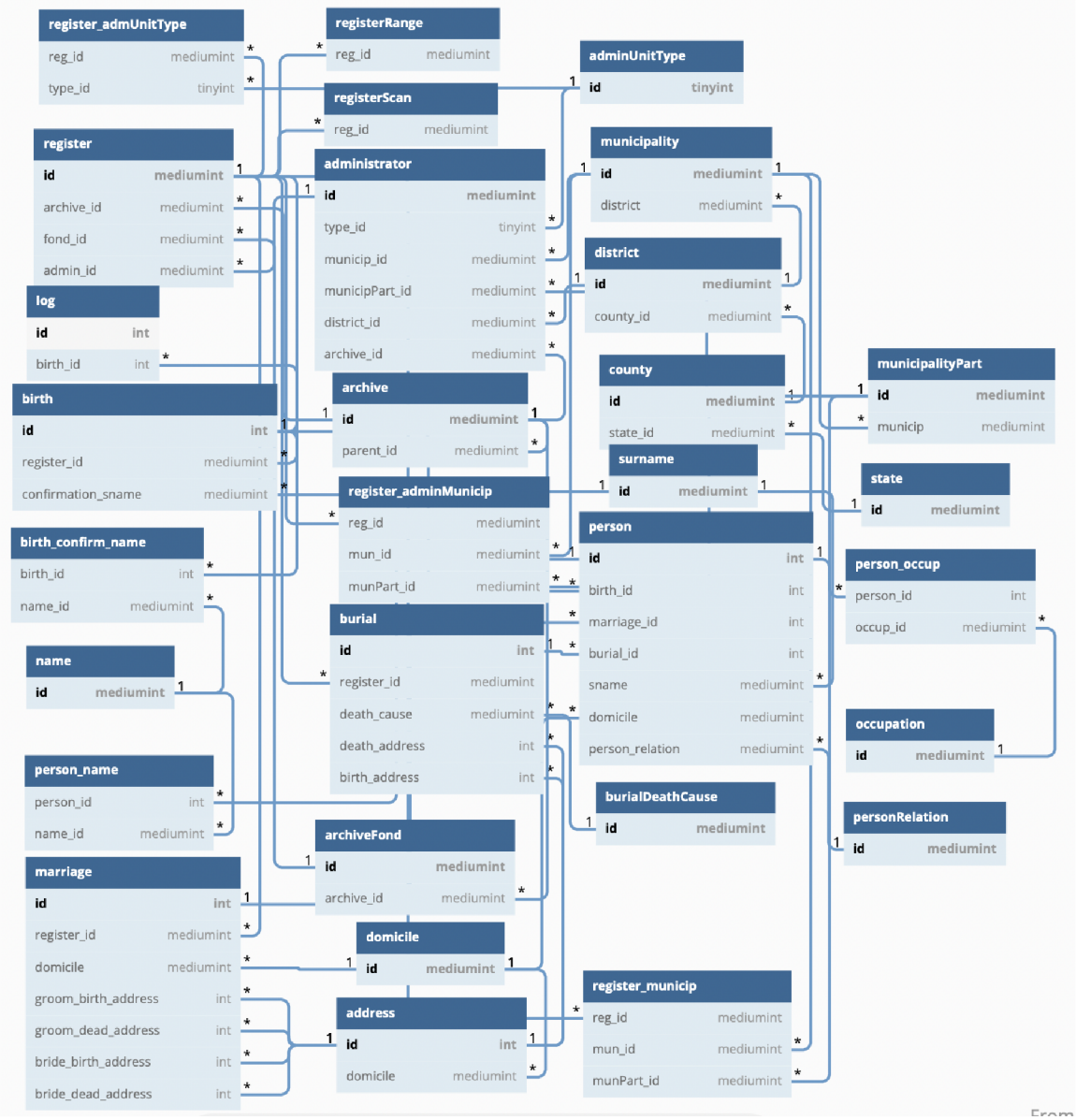



Obr. 3.2: Zjednodušená schéma systémových tabuliek

Druhým typom sú tabuľky, v ktorých sa ukladajú záznamy späté s prepisovaním záznamov z matrík. jednodušenú schému len s primárnym kľúčom a cudzími kľúčmi je možné vidieť na obrázku 3.3. Samotné tabuľky so stručným popisom je možné vidieť v nasledovnom zozname:

- **address** – tabuľka s informáciami o adresách (bydlisko, ulica a popisné číslo)
- **administrator** – tabuľka s informáciami o administračných jednotiek archívov (meno, samospráva, okres, . . . )
- **adminUnitType** – tabuľka s názvami typov administračných jednotiek
- **archive** – tabuľka s informáciami o archívoch (názov, skratka a odkaz na rodičovský archív)
- **archiveFond** – tabuľka obsahujúca informácie o archívnych fondoch (názov, popis a odkaz na archív, do ktorého fond patrí)
- **birth** – tabuľka, ktorá obsahuje informácie o narodení dieťaťa (pohlavie, dátum narodenia, dátum krstu, . . . )
- **burial** – tabuľka s informáciami o pohrebe (dátum úmrtia a pohrebu, miesto úmrtia a pohrebu, . . . )
- **burialDeathCause** – tabuľka s názvami príčin úmrtia
- **county** – tabuľka, ktorá obsahuje zoznam krajov
- **district** – tabuľka obsahujúca zoznam okresov
- **domicile** – tabuľka s názvom sídiel
- **marriage** – tabuľka s informáciami o svadbe (dátum svadby, prípadne aj rozvodu, miesto svadby, vek ženícha a nevesty, . . . )
- **municipality** – tabuľka so samosprávami
- **minicipalityPart** – tabuľka obsahujúca názvy jednotlivých častí samospráv
- **name** – tabuľka s menami osôb
- **occupation** – tabuľka so zamestnaniami
- **person** – tabuľka, ktorá obsahuje informácie o osobách (titul, miesto pracoviska, vieroviznanie, . . . )
- **personRelation** – tabuľka, ktorá obsahuje zoznam vzťahov pre osobu
- **register** – tabuľka obsahujúca informácie o registroch (signatúra, typ registra, počet skenov, . . . )
- **registerRange** – tabuľka, v ktorej sa nachádza typ a časové rozpätie záznamov, ktoré daný register uchováva
- **registerScan** – tabuľka s adresami na vykonané skeny záznamov

- **state** – tabuľka so zoznamom štátov
- **surname** – tabuľka s priezviskami osôb
- a rôzne asociačné tabuľky mapujúce vzťahy medzi databázovými entitami




Obr. 3.3: Zjednodušená schéma tabuliek pre prepis záznamov z matrík

Názvy niektorých entít môžu byť v rôznych tvaroch (napríklad meno Jozef môže by v tvare Jožo alebo Jožko), poprípade inak sa píšu v iných jazykoch alebo sa časom zmenili. Preto je dobré si k názvom a menám priradiť ich normalizovanú podobu. V databáze sa nachádzajú následovné tabuľky pre normalizáciu názvov:

- **normalizedDeathCause** – tabuľka s normalizovanými názvami príčiny úmrtia pre tabuľku `burialDeathCause`
- **normalizedDomicile** – tabuľka obsahujúca normalizované názvy sídiel pre tabuľku `domicile`
- **normalizedName** – tabuľka s normalizovanými menami pre tabuľku `name`
- **normalizedOccupation** – tabuľka obsahujúca normalizované názvy zamestnaní pre tabuľku `occupation`
- **normalizedPersonRelation** – tabuľka obsahujúca normalizované názvy vzťahov pre tabuľku `personRelation`
- **normalizedSurname** – tabuľka obsahujúca normalizované priezviska pre tabuľku `surname`

V databáze sa nachádza aj jedna procedúra s názvom `birthRecord`, ktorá má za úlohu zostaviť kompletný záznam s informáciami o osobách, ktoré sú prepojené s určitým záznamom narodenia. Mimo už spomínaných tabuliek, v databáze sa nachádzajú aj pohľady, ktoré sú využívané spomínanou procedúrou. Jedná sa o:

- **personFull** – pohľad, ktorý získa všetky informácie o osobe
- **birthMarriageFull** – pohľad, ktorý získa všetky informácie spojené so záznamom z tabuľky `birthMarriage`

V pláne je aj integrácia záznamov z iných genealogických zdrojov ako sú matriky (urbáre, lánové registre a iné), pre ktoré bude potreba navrhnúť ďalšie tabuľky. Týmto sa bude zaoberať kapitola 6.

# Kapitola 4

# Analýza databázových systémov

SQL databázy patria podľa stránky DB-Engines[1] k najpopulárnejším databázam. Táto stránka hodnotí popularitu databáz na základe vyhľadávania vo vyhľadávačoch, položených otázkach na stránkach ako Stack Overflow, DBA Stack Exchange a iných faktorov[32]. Na popredných priečkach môžeme vidieť aj NoSQL databázy, z ktorých vyniká MongoDB. Projektu DEMoS potrebuje využívať databázu, ktorá vie hlavne rýchlo a správne vyhľadávať vo veľkých objemoch dát. Keďže záznamy do databázy sa vkladajú manuálne a ich prepis netrvá krátko, tak rýchlosť zápisu nás až tak nemusí zaujímať. Aktuálne sa v databáze nachádza približne 95000 záznamov a predpokladá sa, že v budúcnosti sa tu môže nachádzať až viac ako 125 miliónov záznamov.

V nasledujúcej kapitole popíšem niektoré z SQL a NoSQL databáz a porovnám ich s aktuálne používanou databázou.

## 4.1 SQL databázy

SQL je štandardizovaný programovací jazyk, ktorý je určený na prácu s relačnými databázami, preto relačné databázy označujeme aj ako SQL databázy. Ako bolo už spomenuté, SQL databázy sú najpoužívanejšie databázové systémy. Dôvodom ich dominancie je určite aj to, že relačný model dát je intuitívny a vhodný na použitie pre väčšinu scenárov. Tabuľky relačnej databázy vychádzajú z matematického poňatia relácie, ktoré definuje reláciu ako karteziánsky súčin domén všetkých atribútov.[8]

Keď navrhujeme tabuľky databázy snažíme sa ich vytvoriť tak aby boli efektívne a aby nevznikala redundancia dát. K tomuto nám pomôže takzvaná normalizácia databázových schémat. S pomocou normalizácie si vieme previesť tabuľky z nenormovanej formy do rôzneho stupňa normalizovaných foriem. Existuje 11 úrovní normalizovaných foriem a to 0 – 6 normovaná forma a 4 rozšírenia niektorých z týchto foriem.

Ako som už spomínal, tak systém DEMoS používa MySQL, čo je práve SQL databáza. Existuje množstvo iných relačných databáz, no princíp ich fungovanie je vždy alebo zväčša rovnaký. Niektoré systémy môžu mať navyše funkcie ako MySQL a niektoré môžu pracovať rýchlejšie s dátami. Pri výbere databázového systému je potrebné myslieť nielen na potrebné funkcie, ale aj na cenu licencií. V nasledujúcich podkapitolách zhrniem niektoré vlastnosti a licenčný model databázových systémov MySQL, Oracle Database a PostgreSQL.

[1] https://db-engines.com/en/ranking

### 4.1.1 MySQL

MySQL je relačný databázový systém, ktorý sa aktuálne používa v projekte DEMoS. Tento databázový systém patrí medzi tie najobľúbenejšie aj vďaka tomu, že je multiplatformový a podporovaný väčšinou najpoužívanejších programovacích jazykov. K obľúbenosti určite prispieva aj to, že je zdarma a nie je zložité začať s týmto databázovým systémom. MySQL vie pracovať s veľkými databázami s viac ako päťdesiatimi miliónmi záznamov. Dokonca ako uvádzajú v dokumentácii, tak existujú databázy využívajúce MySQL s viac ako dvesto tisíc tabuľkami a s okolo piatimi miliardami záznamov[24].

MySQL podporuje všetky základné dátové typy ako číselné (`NUMERICAL`, `INTEGER`, `FLOAT` a iné), reťazce (`CHAR`, `VARCHAR`, `TEXT` a iné), časové (`DATE`, `TIME`, `DATETIME` a iné ), ale aj priestorové dáta alebo JSON dokumenty[21].

Niektoré vlastnosti a limity tabuliek môžeme ovplyvniť tým, akým spôsobom povieme databázovému systému, aby ukladal dáta, respektíve podľa toho, aký použijeme takzvaný „Storage Engine“ (ďalej už len SE[2]). MySQL podporuje celú radu SE, ktoré môžeme rôzne kombinovať v našej databáze. Ak pri vytváraní tabuľky neudáme aký chceme použiť SE, tak MySQL použije `InnoDB`. InnoDB a jeho operácie sa riadia modelom ACID, teda podporuje transakčné spracovanie. Ďalej podporuje cudzie kľúče, indexy a iné. Dáta ukladá na disk tak, aby boli dotazy na tieto dáta optimalizované na základe ich primárnych kľúčov. Medzi ďalšie SE, ktoré podporuje MySQL patrí[22]:

- MyISAM
- Memory
- CSV
- Archive
- a iné

MySQL databázový systém môžeme používať bezplatne, využíva GPL open-source licenciu. Ak nemáme v pláne modifikovať zdrojový kód MySQL, tak nás nijak neobmedzuje.

### 4.1.2 Oracle Database

Databázový systém od firmy Oracle, rovnako ako MySQL, patrí k najpopulárnejším databázovým systémom. Je to databázový systém vhodný pre prácu s veľkým množstvom dát, no navyše obsahuje množstvo iných funkcií. Podstatným rozdielom oproti MySQL je, že aj keď Oracle je relačný databázový systém, implementuje rôzne funkcie objektových databáz a preto má označenie ako objektovo-relačný databázový systém. Medzi tieto funkcie patrí napríklad možnosť vytvoriť si vnorenú tabuľku, tvorba vlastného typu, metód s využitím procedurálneho jazyka PL/SQL a iné[35].

Okrem bežných typov, Oracle podporuje aj ukladanie a prácu s priestorovými dátami a grafmi, avšak použitie týchto typov je podporované len v platenej Oracle Enterprise edícii, ktorá bude spomenutá neskôr[6].

Ďalším užitočným typom, ktorý tento databázový systém podporuje je XML dokument. Tieto dokumenty databáza ukladá natívne v objektovom úložisku tak, aby spĺňala W3C DOM špecifikáciu. Okrem toho podporuje XML Schema, XQuery, XPath a XSLT (štýlovací

[2]Nevedel som nájsť vhodný preklad pre tento pojem, tak ďalej budem používať túto skratku

jazyk pre XML). V MySQL je možné pracovať s XML, no natívne ho nepodporuje. Ukladá ho ako typ `BLOB` alebo ako iný z jeho variant a obsahuje rôzne funkcie na prácu s XML[6].

V Oracle Database je možné pracovať aj s multimediálnymi dátami, ako napríklad s obrázkami, zvukom alebo videom a obsahuje rôzne funkcie na manipuláciu s týmito typmi[6].

Okrem spomínaných typov tento databázový systém obsahuje celú radu nástrojov, ktoré sú užitočné na migráciu dát, vývojové nástroje, možnosti využitia paralelizmu alebo databázového firewallu[6].

Nevýhodou tohto databázového systému je jej licencovanie. K dispozícii je Express edícia, ktorá je zdarma, no je dosť okresaná. Aj MySQL je k dispozícii v platených edíciách, no poskytujú navyše len podporu, rýchlejšie aktualizácie, rôzne nástroje na monitorovanie databázy, pokročilé šifrovanie dát a iné[25]. Pri použití Express edície stratíme podstatné množstvo potencionálne užitočných funkcií a takisto sme obmedzený z hľadiska hardvéru (maximálne využíva 11 GB pre dáta, 1 GB RAM a 1 CPU) [26][23]. Preto táto edícia nie je vhodná na nasadenie do produkčného prostredia väčších projektov, ale skôr pre malé projekty a ako základ pre štúdium. Zoznam všetkých funkcií, ktoré obsahuje Express edícia, je možné nájsť na stránkach firmy Oracle[3].

### 4.1.3 PostgreSQL

Ďalšou a poslednou SQL databázou, na ktorú som sa pozrel je PostgeSQL. PostgreSQL je open-source a rovnako ako Oracle Database, objektovo-relačný databázový systém, ktorý si svoje postavenie medzi najpopulárnejšími databázami zaslúžil aj vďaka svojej spoľahlivosti, architektúre, množstva funkcií a vysokému výkonu aj pri použití zložitých dotazov nad touto databázou. Okrem primitívnych dátových typov, vie pracovať aj so štrukturovanými dátovými typmi (dátum a čas, polia, UUID, a iné), dokumentami (JSON, XML, a iné), geometrickými dátami (bod, čiara, kruh, polygon), ale aj s užívateľom vytvorenými typmi[33]. Rovnako ako Oracle, aj Postgres podporuje procedurálny jazyk PL/SQL a niektoré vlastnosti objektových databáz.

PostgreSQL je vyvíjaný pod PostgreSQL Licence, čo je open-source licencia podobná BSD alebo MIT licenciám. Táto licencia nás nijak neobmedzuje z hľadiska používania tohto databázového systému, čo je pre nás veľké plus[34].

Skúsil som si niektoré dotazy nad týmto databázovým systémom, aby som mohol porovnať ako je na tom s časom dotazov v porovnaní s MySQL. Dotazy som vykonával nad databázou vytvorenou z mne poskytnutého dumpu z MySQL databázy systému DEMoS, ktorý som premigroval aj do PostgreSQL. Vykonané dotazy som viackrát zopakoval a následne som si zaznamenal ich priemerný čas. Vykonal som dotazy rôznych náročností, od jednoduchých dotazov typu nájdenia záznamu s určitou hodnotou stĺpca až po zložitejšie dotazy, kde som využil spájanie viacerých tabuliek. Zhrnutie výsledkov týchto dotazov je možné vidieť v tabuľke 4.1. Nájdenie všetkých záznamov adries, ktoré obsahujú stĺpec s názvom ulice „Brandlova“ (dotaz 4.1), trvalo PostgerSQL približne 8 ms, kde MySQL to trvalo 10 ms (tabuľka 4.1 dotaz č. 1). Rozdiel časov týchto dvoch dotazov je zanedbateľný.

```
select *
from address
where street = "Brandlova";
```

Výpis 4.1: Dotaz pre získanie adries obsahujúcich názov ulice „Brandlova“

[3] https://docs.oracle.com/en/database/oracle/oracle-database/18/xelic/licensing-information.html

Väčší rozdiel je pri dotaze, kde chceme získať všetky adresy, ktoré sa nachádzajú v „Lhota“ (dotaz 4.2), kde MySQL to trvalo približne 94 ms a PostgreSQL 20 ms (tabuľka 4.1 dotaz č. 2).

```
select *
from address join domicile on address.domicile = domicile.id
where domicile.name = "Lhota";
```

Výpis 4.2: Dotaz pre získanie adries nachádzajúcich sa v „Lhota“

Tretím príkladom je dotaz, kde chceme získať všetky mená osôb s ich priezviskom (dotaz 4.3), kde MySQL to trvalo približne 689 ms a PostgreSQL 541 ms (tabuľka 4.1 dotaz č. 3).

```
select name.name, surname.name as surname from
(((person join person_name on person_name.person_id = person.id)
join name on name.id = person_name.name_id)
join surname on person.sname = surname.id);
```

Výpis 4.3: Dotaz pre získanie osôb s ich menami a priezviskom

| Č. dotazu | Čas v PostgreSQL [ms] | Čas v MySQL [ms] | Počet navrátených záznamov |
|---|---|---|---|
| 1 | 8 | 10 | 2 |
| 2 | 20 | 58 | 250 |
| 3 | 541 | 689 | 178612 |

Tabuľka 4.1: Testy dotazov vykonaných v PostgreSQL a MySQL

Ako som už spomínal v úvode, tak budem vykonávať aj úpravy v zmysle pridania nových indexov. Indexy vedia výrazne urýchliť čas dotazu. Ako je možné vidieť v tabuľke 4.3 a 4.2, tak pri oboch databázových systémoch s použitím indexov bol čas dotazov o dosť nižší. Indexmi sa bude viac zaoberať kapitola 5.1.

| Č. dotazu | Čas bez indexov [ms] | Čas s indexmi [ms] | Počet navrátených záznamov |
|---|---|---|---|
| 1 | 10 | 1 | 2 |
| 2 | 58 | 27 | 250 |

Tabuľka 4.2: Testy dotazov vykonaných v MySQL s indexmi a bez indexov

| Č. dotazu | Čas bez indexov [ms] | Čas s indexmi [ms] | Počet navrátených záznamov |
|---|---|---|---|
| 1 | 8 | 1 | 2 |
| 2 | 20 | 3 | 250 |

Tabuľka 4.3: Testy dotazov vykonaných v PostgreSQL s indexmi a bez indexov

Z dosiahnutých výsledkov je jasne vidieť, že PostgreSQL dokáže vykonať zvolené dotazy rýchlejšie, aj keď pri niektorých dotazoch je rozdiel zanedbateľný. V budúcnosti, keď systém bude obsahovať viac záznamov a bude citeľne pomalší, by bolo vhodné zvážiť prejdenie na práve tento databázový systém. K migrácii dát by bolo možné použiť open source nástroj PGLoader[4] slúžiaci na načítavanie dát pre tento databázový systém.

[4] https://github.com/dimitri/pgloader

## 4.2 NoSQL databázy

Okrem SQL databáz poznáme aj NoSQL databázy. Význam skratky NoSQL môžeme interpretovať ako „Not Only SQL“, čo nám hovorí, že existujú aj iné modely dát ako relačné a práve databázy patriace do tejto skupiny podporujú prácu s nerelačným modelom dát. NoSQL databázy majú niektoré spoločné vlastnosti, ako napríklad voľná štruktúra atribútov, ktoré sa ukladajú, podporujú distribuovanú architektúru a rýchlosť ako zápisu, tak aj čítania z databázy.

Keďže tieto databázové systémy sú distribuované, vhodné je spomenúť aj takzvaný CAP teorém. CAP teorém, alebo nazývaný tiež Brewerov teorém podla jeho tvorcu Erica Brewera, hovorí o tom, že ak je služba distribuovaná, nemôže byť zároveň konzistentná (**C**onsistency), vždy dostupná (**A**vailability) a odolná voči výpadkom (**P**artition tolerance). Vždy môžu byť splnené maximálne len dve z týchto troch vlastností[5].

Rôzne typy NoSQL databáz si dáta ukladajú v rôznych formátoch a rôznymi spôsobmi. Je to tak preto, lebo sú často optimalizované na rôzne typy úloh. NoSQL databázy môžeme rozdeliť do niekoľkých skupín[37]:

- Kľúč-hodnota NoSQL databázy (Oracle NoSQL)
- Dokumentové databázy (MongoDB)
- Grafové databázy (Neo4j)
- Stĺpcové databázy (Apache Cassandra)

V nasledujúcich podkapitolách popíšem niektoré NoSQL, presnejšie databázy grafové a dokumentové NoSQL databázy.

### 4.2.1 Grafové databázy

Grafové databázy reprezentujú hodnoty ako uzly a vzťahy medzi hodnotami určujú hrany. Známi predstaviteľ grafového databázového systému je napríklad Neo4j. Tento typ databázového systému je vhodné použiť, ak dáta sú navzájom poprepájané, štruktúra dátového modelu nie je konzistentná alebo potrebujeme hlavne čítať dáta[29]. Navyše Neo4j plne podporuje ACID vlastnosti, teda atomičnosť, konzistenciu, izolovanosť a trvácnosť databázových transakcií[14]. Príkladom použitia môže byť napríklad pre ukladanie cestnej, prípadne internetovej siete, pri reprezentácií vzťahov na sociálnych sieťach alebo aj pri ukladaní rodokmeňov. Vhodným využitím grafových databáz je aj analýza vzťahov, čo môže mať určite využitie v systéme DEMoS.

Pri reprezentovaní vzťahov je Neo4j o dosť efektívnejšie, ako napríklad MySQL. Dobrým príkladom je ukladanie priateľov na sociálnej sieti z knihy *Neo4j in Action*[38]. Podobné reprezentácie dát sa môžu vyskytovať v našom systéme, napríklad rodokmene. Pri tomto príklade s použitím MySQL by sme potrebovali dve tabuľky a to tabuľku pre osoby a tabuľku, ktorá ukladá vzťahy. Ak by sme chceli získať všetkých priateľov niektorého používateľa, použili by sme dotaz 4.4[38]. Avšak často by sme chceli si získať aj priateľov priateľa. V tomto prípade pridáme do dotazu ďalšie spájanie tabuliek s pomocou kľúčového slova *join* a takto by sme mohli pokračovať ďalej. Ako autor ďalej uvádza, tak problémom MySQL je, že často sa zaujímame len o konkrétneho používateľa, no vykonávame spájanie celých tabuliek a potom si odfiltrujeme osoby, ktoré nás zaujímajú. Pri veľkom množstve záznamov to môže mať veľký dopad na výkon. V tabuľke 4.4 z knižky *Neo4j in Action* môžeme vidieť

porovnanie dĺžky trvania dotazov, ktoré získavajú rôzne úrovne priateľov v tabuľke s 1000 záznamami o užívateľoch. Pri nízkej úrovni zanorenia sú výsledky prijateľné, no od štvrtej úrovne začína byť tento čas dlhý.

```
select count(distinct uf2.*) from t_user_friend uf1
inner join t_user_friend uf2 on uf1.user_1 = uf2.user_2
where uf1.user_1 = ?
```

Výpis 4.4: Dotaz pre získanie používateľov a ich priateľov v MySQL

| Hĺbka zanorenia | Čas dotazu v sekundách pre 1000 užívateľov | Počet záznamov |
|---|---|---|
| 2 | 0.028 | 900 |
| 3 | 0.213 | 999 |
| 4 | 10.273 | 999 |
| 5 | 92.613 | 999 |

Tabuľka 4.4: Trvanie dotazov pre získanie rôznych úrovní priateľov v MySQL

Vhodnejšia by bola reprezentácia týchto vzťahov v podobe grafu. Keby sme si tieto dáta uložili v Neo4j, rýchlosť operácie získania priateľov by bola o poznanie rýchlejšia. Neo4j využíva koncept traverz grafu (priechod grafom) a použil by sa dotaz 4.5[38].

V tabuľke 4.5 z knižky *Neo4j in Action* môžeme vidieť, že pri použití Neo4j je trvanie dotazu aj pri piatej úrovní zanorenia nízke.

```
TraversalDescription traversalDescription =
    Traversal.description()
    .relationships("IS_FRIEND_OF", Direction.OUTGOING)
    .evaluator(Evaluators.atDepth(5))
    .uniqueness(Uniqueness.NODE_GLOBAL);
Iterable<Node> nodes = traversalDescription.traverse(nodeById).nodes();
```

Výpis 4.5: Dotaz pre získanie používateľov a ich priateľov v Neo4j

| Hĺbka zanorenia | Čas dotazu v sekundách pre 1000 užívateľov | Počet záznamov |
|---|---|---|
| 2 | 0.04 | 900 |
| 3 | 0.06 | 999 |
| 4 | 0.07 | 999 |
| 5 | 0.07 | 999 |

Tabuľka 4.5: Trvanie dotazov pre získanie rôznych úrovní priateľov v Neo4j

Ako môžeme vidieť, tak grafové databázy pracujú efektívnejšie minimálne pri vzťahoch typu Many-To-Many na veľkom počte záznamov ako MySQL. V projekte DEMoS by možno bolo dobré zvážiť použitie tejto databázy pri ukladaní dát, ktoré by boli použité pri reprezentovaní rodokmeňov, respektíve pri reprezentácii genealogických alebo sociálnych modelov.

Neo4j ponúka viacero druhov edícii tejto databázy a to Neo4j Community edition, ktorá je vydávaná pod GPLv3 licenciou a Neo4j Enterprise Edition, ktorá je určená pre komerčné použitie. Komerčnú licenciu je možné získať zdarma pre Startupy alebo aj pre akademické účely[15].

### 4.2.2 Dokumentové databázy

Populárnou NoSQL databázou je aj MongoDB, čo je databázový systém, ktorý si ukladá dáta ako dokumenty. Tieto dokumenty sú uložené vo formáte BSON (Binary JSON), ktorý je založený na formáte JSON. Dokumenty v tomto databázovom systéme nemajú pevne zadefinovanú štruktúru jednotlivých atribútov, a to či už názvy atribútov, tak aj ich typy. Táto vlastnosť dokumentových databáz sa hodí pri vývoji aplikácie, ak štruktúra dát nie je ešte pevne daná a môže sa meniť. Avšak tento prístup má aj svoje nevýhody a to je konzistencia dát. Výhodou dokumentových databáz je aj to, že samotné dokumenty sú ľahko čitateľné aj bez ďalšieho spracovania, môžeme tam uložiť nielen štrukturované dáta, ale ja tie neštrukturované (textové dokumenty, obrázky, videá, . . . ) [11].

Silnou stránkou relačných databáz je ich dotazovací jazyk, ktorého sila spočíva v prieskume dát. Dotazovací jazyk MongoDB je naopak zameraný na vývojárov aplikácii a využíva notácie jazyka JavaScript. Dokumentové databázy, na rozdiel od relačných databáz, neobsahujú cudzie kľúče, vzťahy one-to-many a one-to-one riešia vnorenými dokumentami. Takéto dokumenty niesu normalizované, čo nám pri NoSQL databázach zas až tak nevadí. Avšak je možné využiť aj hodnoty atribútov ako kľúč do iného dokumentu a s využitím agregácie si môžeme odkazovaný dokument vložiť ako vnorený dokument. Táto operácia je ale dosť drahá. Druhou možnosťou, ktorú môžeme využiť, ak potrebujeme mať oddelené dokumenty a chceme sa medzi nimi odkazovať, je použitie referencií. Referencie fungujú ako ukazovateľ do druhej tabuľky na základe takzvaného „ObjectID“, ktorý sa nachádza v každom dokumente v atribúte `_id`.

Ak by sme chceli prejsť na tento databázový systém, museli by sme zmeniť štruktúru tabuliek, respektíve by sme museli denormalizovať tabuľky (čo pri NoSQL databázach nám nevadí), keďže MongoDB nepodporuje prácu s cudzími kľúčmi. Ak by sme namapovali tabuľky na dokumenty bez zmien, mohli by sme využiť agregáciu, čo je avšak veľmi drahá operácia, dotazy by boli zložité a trvali by dosť času. Avšak pri dotazovaní nad samotnými dokumentami je rýchle. Druhou možnosťou by bolo využívať MongoDB len na čítanie, zápis a samotnú prácu s dátami by sme mohli ponechať na aplikáciu bežiacu na serveri.

Z hľadiska licencovanie, MongoDB je od 16. októbra 2018 vydávané pod licenciou Server Side Public Licence. Ak je databáza používaná len ako úložisko a nie ako služba, tak nás táto licencia nijak neobmedzuje a môžeme MongoDB bez problému používať[10].

Skúšal som vykonať niektoré dotazy nad touto databázou, do ktorej som premigroval tabuľky z MySQL s pomocou knižníc Sequlize a mysql2. S pomocou týchto knižníc som si vyexportoval dáta z tabuliek do JSON dokumentov a následne som ich importoval do MongoDB databázy s pomocou aplikácie MongoDB Compass. Dotaz, pomocou ktorého som si chcel zistiť všetky adresy, ktoré obsahovali názov ulice „Brandlova“ (tabuľka 4.6 dotaz č. 1) prebehol relatívne rýchlo (14 ms), aj keď to MongoDB trvalo dlhšie ako MySQL (10 ms). Použitý dotaz 4.6.

```
db.address.find({street: "Brandlova"})
```

Výpis 4.6: Dotaz pre získanie adries obsahujúcich ulicu „Brandlova“ v MongoDB

Horšie na tom boli dotazy, kde som využíval agregáciu. Pri zadaní dotazu, v ktorom som chcel zistiť všetky ulice pre záznam obce s hodnotou „Lhota“ (tabuľka 4.6 dotaz č. 2) to na mojom hardvéri trvalo až 183892 ms, no popritom MySQL to trvalo len 58 ms. Použitý bol dotaz 4.7.

```
db.address.aggregate([{
    $lookup: {
        from: "domicile",
        localField: "domicile",
        foreignField: "id",
        as: "domicile",
        pipeline: [{
            $match: {
                $expr: {
                    $eq: ["$name", "Lhota"]
                }
            }
        }]
    }
}, {
    $match: {
        domicile: {
            $not: {
                $size: 0
            }
        }
    }
}])
```

Výpis 4.7: Dotaz pre získanie ulíc nachádzajúcich sa v „Lhota“

| Č. dotazu | Čas v MongoDB | Čas v MySQL | Počet navrátených záznamov |
|---|---|---|---|
| 1 | 14 | 10 | 2 |
| 2 | 183892 | 58 | 250 |

Tabuľka 4.6: Testy dotazov vykonaných v MongoDB a MySQL

## 4.3 Zhrnutie

Systém DEMoS aktuálne využíva relačný databázový systém MySQL. Tento databázový systém má vhodné vlastnosti pre potreby systému, teda vie pracovať a rýchlo vyhľadávať vo veľkom objeme dát. S využitím indexov je vyhľadávanie porovnateľné s inými relačnými databázami, rozdiely sú minimálne.

Výnimkou by mohli byť dotazy, ktoré využívajú viacero spájaní tabuliek (dotazy popísané v príklade v sekcii 4.2.1), kde pri vyššom počte záznamov by bol čas MySQL dotazu neprijateľný a vhodnou by bola grafová databáza (napríklad Neo4j). Grafové databázy by sa mohli využiť napríklad pre ukladanie dát, ktoré by boli využité rodokmeňoch. Avšak pri ukladaní ostatných informácií o systéme (užívatelia, štáty, okresy, ...) alebo pri ukladaní samotných prepísaných záznamov by to už nedávalo až taký zmysel.

Čo sa týka ostatných spomínaných relačných databázových systémov, tak ponúkajú viac-menej to isté ako MySQL (u Oracle DB je potrebná kúpa licencie) a aj podobnú rýchlosť vyhľadávania. Použitie dokumentových databázových systémov by nám mohlo priniesť rôzne výhody, ako napríklad voľnú štruktúru dát alebo možnosť ukladať dáta distribuované, no bolo by potrebné a vhodné vykonať pomerne veľký zásah ako do systému, tak aj do návrhu ukladania dát. Z tohoto vyplýva, že prínos zmeny databázového systému, aspoň v aktuálnom stave databázy, by bol menší ako cena vykonania týchto zmien.

# Kapitola 5

# Revízia aktuálnej databázovej schémy

## 5.1 Úprava schémy

Návrh a implementácia databázy systému nevznikla naraz, ale postupne sa pridávali nové tabuľky podľa potreby. Z tohto dôvodu mohlo pri vytváraní nových tabuliek dôjsť k vzniku rôznych chýb, napríklad pri normalizovaní tabuliek sa mohlo zabudnúť na vymazanie niektorých stĺpcov, ktoré boli nahradené tabuľkami. Napríklad v tabuľke `person` sa nachádzajú stĺpce `domicile`, `street` a `descr_num`, ktoré zaznamenávajú adresu. Tieto stĺpce by sa mohli nahradiť jedným stĺpcom, ktorý by obsahoval cudzí kľúč do už existujúcej tabuľky adries. Rovnako to je s tabuľkou `birthMarriage`, ktorá obsahuje znova tieto stĺpce a rovnako by mohli byť nahradené cudzím kľúčom do tabuľky adries.

### Normalizácia

Niektoré tabuľky je možné ešte normalizovať. Napríklad tabuľka `register` obsahuje stĺpce `lang1`, `lang2` a `lang3`, ktoré môžu byť nahradené tabuľkou s jazykmi a tabuľkou, ktorá by mapovala M:N vzťahy. Aj keď je to asi málo pravdepodobné, tak by sa mohlo stať, že by sme chceli upraviť niektorý z jazykov a museli by sme ho upraviť v každom zázname v tejto tabuľke. Pri použití samostatnej tabuľky s jazykmi a vzťahovej tabuľky by sme v tomto prípade ušetrili dosť času. Takisto by sa v tomto prípade odstránil stĺpec `lang` v tabuľke `birth`, `burial` a `marriage` a nahradil ich odkazmi do vzťahovej tabuľky s jazykmi pre tieto tabuľky. Podobnú normalizáciu by bolo možné vykonať v tabuľke `marriage` pre stĺpce `banns1`, `banns2` a `banns3` s tým, že tabuľka `banns` by obsahovala mimo stĺpca pre dátum aj stĺpec pre poradie.

### Typy atribútov

Pri ukladaní záznamov je potrebné myslieť na typy jednotlivých stĺpcov. Kvôli zle určenému dátovému typu stĺpca môžeme prísť pri veľkom počte záznamov o výkon, prípadne samotné záznamy môžu zaberať zbytočne veľa priestoru v pamäti.

V tabuľke `archive` sa nachádza stĺpec `name`, ktorý je typu `TEXT`. Použitie tohto typu má niektoré výhody, napríklad nie je potrebné si určiť maximálnu dĺžku reťazca, čo môže byť zároveň aj jeho nevýhodou, no zároveň stĺpec s týmto typom nemôže byť celý použitý ako index. Ak by sme chceli použiť stĺpec typu `TEXT` ako index, museli by sme si určiť jeho

prefix. Druhou vecou je spôsob, akým je stĺpec ukladaný. Pri použití tohoto typu dáta nie sú ukladané priamo v tabuľke, uložené sú na disku a v stĺpci je uložený len ukazovateľ. Pri vykonávaní dotazu nad týmto stĺpcom sa obsah nahrá do dočasnej tabuľky na disku, čo spôsobí zníženie výkonu[19]. Keďže dáta sa ukladajú na disk, tak v samotnej tabuľke zaberajú menej miesta. Preto je vhodné tento typ použiť ak máme tabuľky s veľkým množstvom stĺpcov, kde samotný záznam zaberá pomerne dosť miesta[1] alebo pri ukladaní dlhých textov, ako napríklad celé odseky nejakého textu. Z týchto dôvodov by som zmenil typ stĺpca `name` v tejto tabuľke na typ `VARCHAR`, kde by som nastavil maximálnu dĺžku na 100 znakov. Na maximálnu dĺžku som prišiel tak, že som si našiel s pomocou SQL príkazu najdlhšie meno archívu (v našom prípade je to 67 znakov), ktorý je v tabuľke a zaokrúhlil som jeho dĺžku na stovky, aby sme mali ešte rezervu pre dlhšie názvy. Použil som nasledujúci dotaz 5.1, ktorý je možné vidieť nižšie.

```
select length(name)
from archive
order by length(name) desc
limit 1;
```

Výpis 5.1: Dotaz pre získanie najdlhšieho názvu archívu

Podobne by sme mohli zmeniť tento typ na `VARCHAR` aj v ostatných tabuľkách, no v tomto prípade by nám nešlo o tvorbu indexov, ale len o zvýšenie výkonu pri vykonávaní dotazu. V nasledujúcom zozname môžeme vidieť zoznam tabuliek a stĺpce, ktorých by bolo možné zmeniť tento typ[2]:

- **person** – stĺpec `work_place` na typ `VARCHAR(100)`
- **burial** – stĺpce `death_place`, `burial_place` a `marriage_place` na typ `VARCHAR(100)` a stĺpec `dead_time` na `VARCHAR(25)`

Druhov vecou, nad ktorou je potrebné sa zamyslieť pri našej databáze je, či definícia tabuľky spĺňa požadované vlastnosti systému, presnejšie vlastnosť zachovania pôvodnej podoby predpisovaného záznamu, ktorý môže obsahovať chyby alebo jeho časť môže byť nečitateľná. Túto vlastnosť môžeme docieliť tým, že niektoré stĺpce budeme ukladať ako reťazce. V reťazci potom môžeme vyznačiť napríklad nečitateľný dátum ako „??-??-1825“. V našej databáze je táto vlastnosť zaistená s použitím spomínaného dátového typu reťazec napríklad v stĺpcoch pre rôzne dátumy.

### Indexy

Indexy v databázových systémoch sa používajú na zrýchlenie vyhľadávania v tabuľkách. Indexy môžeme chápať ako ukazovatele na riadky v tabuľke. Tieto ukazovatele sú uložené v špeciálnych vyhľadávacích tabuľkách, kde typ tejto tabuľky môže byť rôzny pre rôzne databázové systémy. MySQL používa B-strom pre ukladanie väčšiny indexov, výnimkou sú len indexy pre priestorové dáta, ktoré sú uložené v R-strome. Pre primárny kľúč je v MySQL automaticky vytvorený index, pre ostatné stĺpce sa indexy vytvárajú s pomocou príkazu `CREATE INDEX názovIndexu on názovTabuľky (názovStĺpca, ...)`[20]. Indexy majú aj svoje nevýhody a to, že zaberajú miesto a spomaľujú vkladanie nových záznamov do tabuľky. Spomalené vkladanie nám pre väčšinu tabuliek, ktoré spomeniem nižšie, nebude

[1] Maximálna dĺžka stĺpca v MySQL je 65 535 bytov

[2] Pri nastavovaní maximálnej dĺžky reťazca som postupoval podobne ako v predchádzajúcom odseku

vadiť, lebo sa bude jednať o viac-menej tabuľky so záznamami, ktoré sa moc nemenia (názvy obcí, štátov a iné).

V našej databáze sa okrem indexov pre primárne kľúče nachádzajú len indexy pre cudzie kľúče do iných tabuliek. Po analýze tabuliek a systému by som navrhoval pridať indexy pre niektoré tabuľky. Tieto atribúty majú potenciál na to, aby sa vyskytovali v rôznych dotazoch a mohli by sa použiť v rôznych vyhľadávaniach. Príkladom môže byť meno alebo priezvisko osoby, podľa ktorých by sa mohli vyhľadávať záznamy pre túto osobu. Je možné, že v aktuálnom stave systému by sa všetky mnou navrhnuté indexy nevyužili, no v budúcnosti keď systém bude podporovať viac zložitejších možností vyhľadávania, mohli by sa hodiť. Vytvoril som indexy pre nasledujúce tabuľky so stĺpcami nachádzajúcimi sa v nasledujúcom zozname:

- **address** – pre stĺpce *street*, *descr_num*
- **administrator** – pre stĺpec *name*
- **archive** – pre stĺpce *name*, *short*
- **archiveFond** – pre stĺpec *name*
- **birth** – pre stĺpce *baptism_date*, *birth_date*
- **burial** – pre stĺpce *death_date*, *burial_date*, *burial_place*
- **burialDeathCause** – pre stĺpec *name*
- **county** – pre stĺpec *name*
- **district** – pre stĺpec *name*
- **domicile** – pre stĺpec *name*
- **log** – pre stĺpec *datetime*
- **marriage** – pre stĺpec *marriage_date*
- **municipality** – pre stĺpec *name*
- **municipalityPart** – pre stĺpec *name*
- **name** – pre stĺpec *name*
- **person** – pre stĺpce *title*, *work_place*
- **register** – pre stĺpce *signature*
- **state** – pre stĺpec *name*
- **surname** – pre stĺpec *name*
- **user** – pre stĺpce *username*, *email*

Vykonal som niekoľko dotazov na našou databázou, aby som porovnal to, ako moc zrýchlia vyhľadávanie niektoré spomenuté indexy zo zoznamu vyššie. V tabuľke 5.1 je možné vidieť výsledky vykonaných testov. Použité boli dotazy rovnaké, ako v kapitole 4.1.3.

| Č. dotazu | Čas bez indexov [ms] | Čas s indexmi [ms] | Počet navrátených záznamov |
|---|---|---|---|
| 1 | 8 | TODO | 900 |
| 2 | 20 | TODO | 999 |
| 3 | 541 | TODO | 999 |

Tabuľka 5.1: Testy dotazov vykonaných v MySQL s indexmi a bez indexov

## 5.2 Procedúra na získanie kompletného záznamu o osobe

V databázovom systéme sa nachádza procedúra s názvom `birthRecord`, ktorá má za úlohu získať kompletný záznam o narodení osoby. Vstupom procedúry je primárny kľúč záznamu o narodení. Táto procedúra môže mať využitie pre niekoho, komu sa lepšie pracuje napríklad v tabuľkových procesoroch (MS Excel, atď.) a chce si vyexportovať všetky tieto údaje do formátu CSV. Keďže procedúra vracia všetky informácie o narodení konkrétnej osoby, vrátane informácie o osobách, ktoré sú v nejakom vzťahu s touto osobou, je potrebné zostaviť výsledok pospájaním rôznych tabuliek. Čas získania výsledku tejto procedúry je dosť dlhý (v mojom prípade okolo minúty), tak som sa pozrel na možné spôsoby urýchlenia.

Problém nastáva pri používaní pohľadu, ktorý nám získa všetky informácie o konkrétnej osobe. Každý tento pohľad pristupuje k rôznym tabuľkám (tabuľky o osobách, priezviskách, bydlísk, zamestnaní, vzťahov a mien). Všetky tieto tabuľky sú pospájané na základe primárnych a cudzích kľúčov, čo je časovo náročná operácia. Pri volaní procedúry sa musí zostaviť až 22 pohľadov pre osobu, čo zaberá veľa času. Dôvodom prečo to tak dlho trvá je, že každý pohľad zostavuje tieto informácie pri každom volaní nanovo. Z tohto dôvodu je potrebné sa zamyslieť, či sa nedá nejakým spôsobom obmedziť zostavovanie týchto informácií na minimum.

Možným riešením by mohlo byť použitie dočasných tabuliek namiesto pohľadov, do ktorých by sa na začiatku procedúry uložili všetky potrebné informácie o osobách. Vďaka použitiu takejto tabuľky by sa ušetrilo veľa času, keďže sa nevytvára nový požiadavok pre každú osobu, ale len sa vyhľadá záznam v tejto tabuľke pre danú osobu. Takto sa čas potrebný na vykonanie procedúry zmenší na menej ako jednu sekundu.

Avšak problémom je, že MySQL nepodporuje viacnásobné čítanie z jednej dočasnej tabuľky. Možným riešením by bol prechod na databázový systém MariaDB (iná populárna open-source verzia MySQL), ktorá túto vlastnosť podporuje. Ak sa ale neprejde na tento databázový systém, musí existovať iné riešenie tohto problému.

Riešením, na ktoré som prišiel je, že na začiatku procedúry sa vytvorí tabuľka s názvom `TEMP_personFull`, ktorá nahradí doteraz používaný pohľad s názvom `personFull`. Tabuľka obsahuje štruktúru stĺpcov, ktorá je zhodná so spomínaným pohľadom. Následne sa novo vytvorená tabuľka naplní informáciami o všetkých osobách, ktoré sú v spojení z daným záznamom o narodení. Potom pri zostavení kompletného záznamu o narodení sa na miestach, kde bol použitý spomínaný pohľad, použije práve táto novo vytvorená tabuľka. Poslednou vecou, ktorú je potrebné vykonať je zmazanie tejto tabuľky na konci procedúry.

## 5.3 Chýbajúci typ registra

Každý historický zdroj sa ukladá do registra, ktorý je určený pre daný historický zdroj. Typ registra určuje stĺpec „type“, ktorý je typu `enum('MAT','LR','RA','URB','URBOP')`. Problémom je, že tu chýba typ pre historický prameň „Berní rule“. Z tohto dôvodu som pozmenil typ tohto stĺpca pridaním nového typu „BR“. Na vykonanie tejto úpravy bol použitý dotaz 5.2.

```
ALTER TABLE ‘tacr‘.‘register‘
CHANGE COLUMN ‘type‘ ‘type‘ ENUM('MAT', 'LR', 'RA', 'URB', 'URBOP', 'BR') CHARACTER
    SET 'utf8' COLLATE 'utf8_bin' NULL DEFAULT NULL ;
```

Výpis 5.2: Dotaz pre úpravu typu stĺpca v tabuľke pre registre

# Kapitola 6

# Integrácia ďalších genealogických zdrojov

V databáze DEMoS sa aktuálne nachádzajú len záznamy, ktoré pochádzajú z matrík. Tento projekt si za cieľ stanovuje vytvoriť databázu, ktorá bude umožňovať prepis nie len z matrík, ale aj z iných historických prameňov ako sú napríklad pozemkové knihy. Celkovo bude potrebné integrovať do systému ďalšie štyri historické pramene, a to „Urbáře“, „Lánové rejstříky“, „Poddanská přiznávací fase“ a „Berní rula“. Pre tieto záznamy je potrebné navrhnúť vhodnú schému databázových tabuliek, čím sa bude zaoberať nasledujúca kapitola. Vychádzať budem zo schém týchto historických zdrojov, ktoré mi boli poskytnuté.

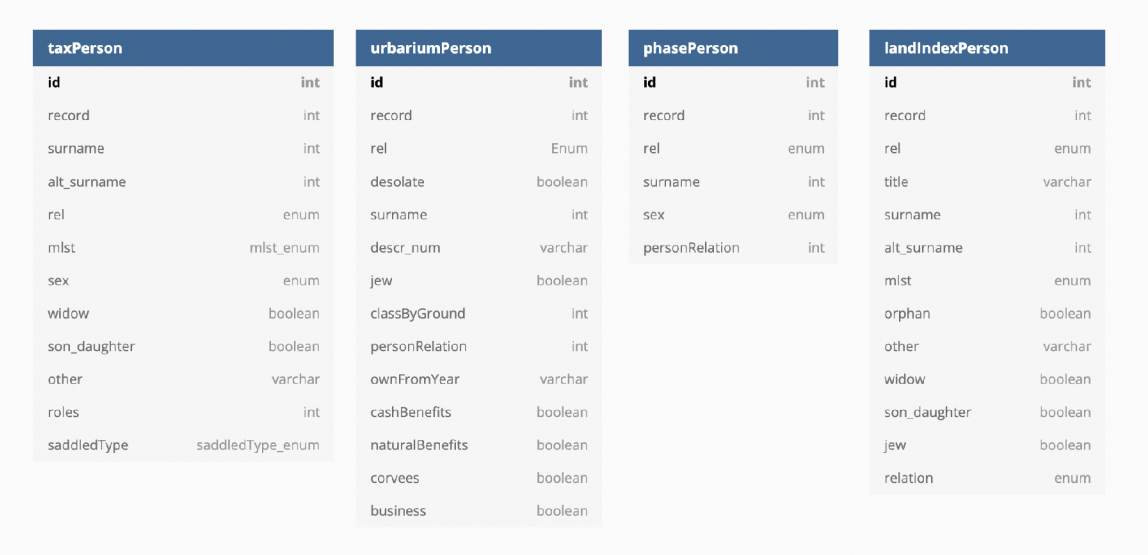
## 6.1 Tabuľka s osobami

Ešte pred tvorbou samotných tabuliek sa treba pozrieť na jeden z problémov. Treba sa zamyslieť nad tým, ako budeme ukladať záznamy o osobách, ktoré sú späté s daným záznamom. Problémom je to, že pri jednotlivých historických prameňoch sa vedú iné informácie o osobe. Máme tri možnosti, a to buď doplníme aktuálnu tabuľku s osobami o chýbajúce atribúty, vytvoríme novú tabuľku so všetkými atribútmi osôb z nových zdrojov alebo pre každý historický prameň vytvoríme samostatnú tabuľku s informáciami o osobách.

Prvá možnosť by bola jednoducho implementovateľná, nevznikali by redundantné väzby medzi osobou a menom, priezviskom a inými tabuľkami. Avšak by tento záznam obsahoval veľa nevyplnených atribútov, keďže veľa stĺpcov by bolo irelevantných k niektorým záznamom. Dobrým ukazovateľom by bol prienik osôb záznamov z matrík s osobou z týchto nových historických prameňov. Prienik týchto tabuliek nie je stopercentný. Príkladom môže byť prienik aktuálnej tabuľky s osobami a s tabuľkou s osobami zo záznamov z urbárov, kde pri niektorých úpravách ( napríklad nahradením stĺpca `jew` stĺpcom `religion` z aktuálnej tabuľky s osobami, keďže už obsahuje túto hodnotu), majú spoločných len 5 stĺpcov. Druhou nevýhodou by mohla byť zložitejšia úprava štruktúry týchto zdrojov. Pri použití tejto možnosti, by sme takisto vedeli priradiť viacero typov záznamov k jednej osobe (napríklad záznam z matriky a z urbáru). Problémom ale je, že nevieme s istotou určiť, či osoba, s ktorou sú prepojené záznamy, je tá istá osoba. S podobnými problémami sa môžeme stretnú pri použití druhej možnosti. Jedinou výnimkou by bolo zavedenie pár redundantných väzieb.

Tretia možnosť by síce zaviedla redundanciu niektorých vzťahov, no prinieslo by nám to aj niektoré výhody. Výhodou môže byť jednoduchšie dotazovanie, lepšia normalizácia tabuliek, ale aj by bolo možné jednoduchšie vykonať úpravy štruktúry zdrojov. Z týchto

dôvodov by bola vhodnejšia práve táto tretia možnosť. Samotné schémy tabuliek pre jednotlivé osoby je možné vidieť na obrázku 6.1.




Obr. 6.1: Schémy tabuliek osôb pre jednotlivé historické pramene

## 6.2 Urbáře

Urbár je historický prameň, ktorý slúžil na spísanie pozemkového majetku a povinnosti poddaných. Vznikol pre evidenciu dávok a platieb, ktoré dostávali majitelia panstiev[36]. V našej databáze bude potrebné pre prepis záznamov z urbárov vytvoriť päť nových tabuliek. Schémy všetkých týchto tabuliek, spolu s už existujúcimi tabuľkami, je možné vidieť nižšie na obrázku 6.2.

Prvá tabuľka bude reprezentovať samotný prepísaný záznam, `urbariumRecord`. Mimo informácií, ktoré sú využívané systémom DEMoS (autor prepisu, získané skóre, level zapisovateľa a iné) sa v tejto tabuľke budú nachádzať informácie o samotnom zázname (dátum vytvorenia, jazyk záznamu, odkaz na register, sériové číslo, názov obce a iné).

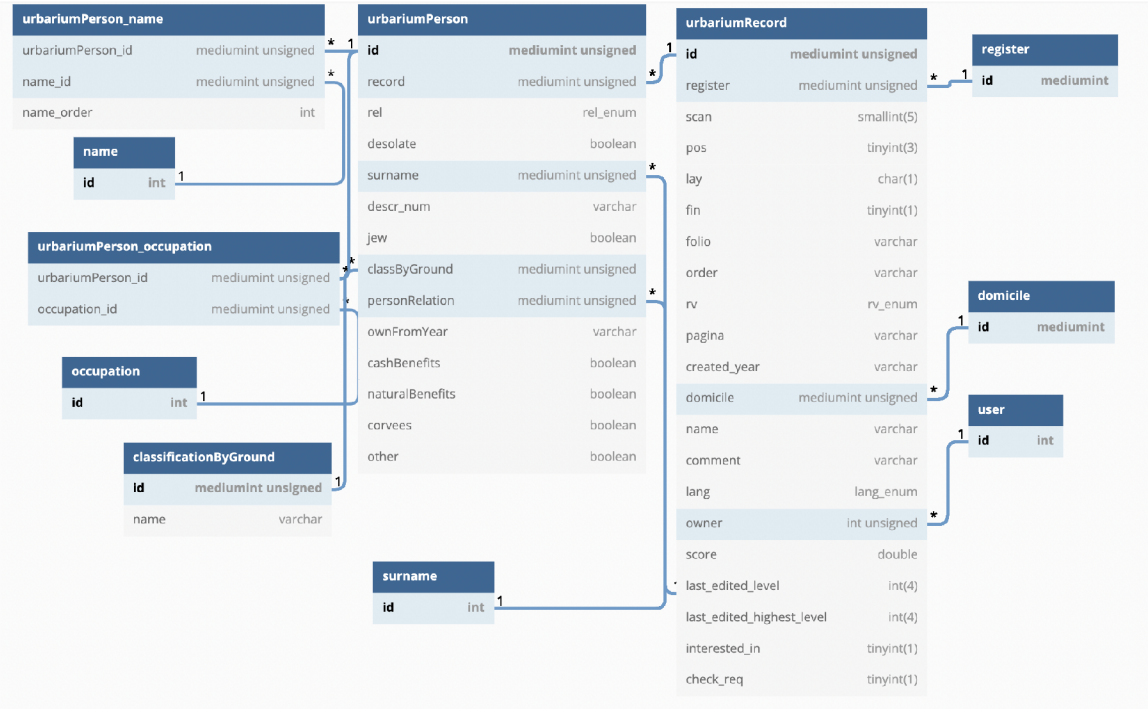
Druhá tabuľka, `urbariumPerson`, bude obsahovať dáta o osobe, s ktorou je daný záznam spätý. Budú sa tu nachádzať informácie ako bydlisko a zázemie osoby, rok od kedy daná osoba vlastní pozemok, či využíva daný pozemok na podnikanie, a iné. Takisto sa v systéme budú viesť záznamy nielen o pôvodnom vlastníkovi, ale aj o nasledujúcich vlastníkoch (ak existovali). Aby sme vedeli rozlíšiť jednotlivé osoby, je zavedený stĺpec `rel`, ktorý nám hovorí o aký typ osoby sa jedná. V databáze bude možné uložiť nasledujúce typy osôb pre urbáre:

- `urb_orig` – označuje pôvodného vlastníka
- `urb_orig_rel` – označuje osobu, s ktorou bol pôvodný vlastník v nejakom vzťahu, ktorý je určený stĺpcom `personRelation`
- `urb_next_1, urb_next_2, urb_next_3` – označuje nesledujúcich vlastníkov, v databáze bude možné zaznamenávať až ďalších troch

- `urb_next_1_rel, urb_next_2_rel, urb_next_3_rel` – označuje osoby, s ktorými bol nasledujúci vlastník v nejakom vzťahu, ktorý je, rovnako pri pôvodnom vlastníkovi, určený stĺpcom `personRelation`

Rovnaký stĺpec (samozrejme s inými hodnotami) je zavedený aj v tabuľkách s osobami pre ďalšie historické pramene, ktoré popíšem neskôr.

Mimo spomínaných tabuliek bude potrebné vytvoriť aj tabuľky obsahujúce vzťah medzi novou tabuľkou s osobou a menom a zamestnaním. A nakoniec v rámci normalizácie sa ešte vytvorí tabuľka, ktorá bude obsahovať názvy zázemí pre osobu. Pre ukladanie informácií o bydlisku alebo registri, z ktorého pochádza záznam, budú využité doposiaľ vytvorené tabuľky, ktoré sa využívajú na ukladanie záznamov z matrík, a to tabuľky `register`, `name`, `occupation`, `domicile` a `user`.




Obr. 6.2: Schémy tabuliek pre záznamy z urbáru

### Popis niektorých stĺpcov tabuľky urbariumRecord

- `created_year` – rok spísania
- `name` – názov gruntu

**Popis niektorých stĺpcov tabuľky urbariumPerson**

- `desolate` – označuje, či grunt bol opustený
- `descr_num` – popisné číslo
- `ownFromYear` – rok, od ktorého je osoba vlastníkom
- `jew` – označuje, či je daná osoba židom
- `cashBenefits` – označuje, peňažné dávky
- `naturalBenefits` – označuje, naturálne benefity
- `corvees` – označuje, či boli vykonávané práce (na pozmeko patriace pánovi)
- `other` – ostatné informácie

## 6.3 Lánové rejstříky

„Lánový rejstřík“ je najstarší moravský kataster poddanských usadlostí a pozemkov. Využívaný bol na zmapovanie území, aby bolo možné ľahšie určiť výšku daní obyvateľstva[41]. Pre tento historický prameň bude potrebné v našej databáze vytvoriť hneď niekoľko nových tabuliek. Schémy všetkých týchto tabuliek, spolu s už existujúcimi tabuľkami, je možné vidieť nižšie na obrázku 6.3.

Ako aj pri urbároch, tak aj tu bude potrebná tabuľka pre samotný záznam s názvom `landIndexRecord`. Táto tabuľka okrem informácií využívaných systémom (rovnako ako pri urbároch) bude obsahovať atribúty, ktoré budú popisovať samotný pozemok. Tieto atribúty udávajú, nielen či sa na pozemku nachádzali vinice, pole, mlyn, dom pre sluhov a iné ale aj umiestnenie pozemku.

Rovnako druhou tabuľkou bude tabuľka s informáciami o osobe spojenou s daným záznamom, `landIndexPerson`. Obsahovať bude informácie o držiteľovi gruntu (meno, priezvisko, či daná osoba je sirota, ak existovala osoba s rovnakým menom, tak atribút mladší alebo starší a iné) a základné informácie (meno a priezvisko) o rôznych osobách v príbuzenskom vzťahu s touto osobou. V databáze bude možné uložiť nasledujúce typy osôb:

- `li_main` – hlavná osoba záznamu
- `li_orphan` – osoba, po ktorej bola hlavná osoba sirota
- `li_widow` – osoba, po ktorej bola hlavná osoba vdova/vdovec
- `li_child` – osoba, koho bola rodičom hlavnej osoby
- `li_fo, li_of` – vlastníci polí
- `li_vo` – osoba, ktorá predstavuje držiteľa vinohradu

Ďalšími tabuľkami budú tabuľka, v ktorej budú informácie o poliach a tabuľka o vinohradoch. V tabuľke o poliach mimo vlastníka bude ešte obsahovať informácie ako rozmery, či je dané pole voľné alebo jeho typ. Pri viniciach sa vedú informácie o rozmeroch a názov vinice.

Aj pre „Lánové rejstříky“ bude potrebné vytvoriť tabuľky pre vzťahy medzi dvoma entitami. Konkrétne sa jedná o tabuľku, ktorá bude prepájať osobu na jej mená. Ostatné potrebné tabuľky na vytvorenie sú tabuľka s názvom sídla a tabuľka s druhmi živností.

Rovnako ako pri záznamoch z urbárov, tak aj tu bude možné využiť už existujúce tabuľky. Patria medzi ne tabuľky `registry`, `domicile`, `classificationByGround`, `user`, `name` a `surname`.


Obr. 6.3: Schémy tabuliek pre záznamy zo zdroja „Lánové rejstříky“

### Popis niektorých stĺpcov tabuľky vineyard

- `area_merice` – rozloha v mericiach
- `area_achtel` – rozloha v achtelch
- `sown_area_merice` – rozloha zasiateho viniča v mericiach
- `sown_area_achtel` – rozloha zasiateho viniča v achtloch
- `name` – názov viničnej trate

**Popis niektorých stĺpcov tabuľky landIndexPerson**

- `title` – titul osoby
- `surname` – priezvisko osoby
- `alt_surname` – alternatívne priezvisko osoby
- `mlst` – označuje ak existovala osoba s rovnakým menom, že bol mladší alebo starší
- `orphan` – označuje, či osoba bola sirota
- `widow` – označuje, či osoba bola vdova/vdovec
- `son_daughter` – označuje, či osoba bola syn, či dcéra niekoho
- `jew` – označuje, či osoba bola židom
- `sex` – pohlavie osoby
- `other` – ostatné informácie

**Popis niektorých stĺpcov tabuľky landIndexRecord**

- `manor` – názov panstva
- `domicile` – názov obce
- `landName` – názov gruntu
- `ground_category` – názov kategórie zeme
- `tradeType` – názov druhu živnosti
- `vineyard` – informácie o vinohrade
- `field` – informácie o poli
- `other_field` – informácie o inom poli
- `servant` – šenkovský mešťan
- `servant_house` – názov šenkovského domu
- `miller` – označuje pôsobenie mlynára
- `mill` – označuje, či sa tu nachádzal mlyn
- `desolation_year` – rok spustnutia usadlosti
- `desolation_reason` – dôvod spustnutia usadlosti
- `construction_year` – rok zbudovania usadlosti
- `take_year` – rok prevzatia usadlosti

**Popis niektorých stĺpcov tabuľky field**

- `free` – označuje, či je pole voľné
- `count` – počet polí
- `area_merice` – rozloha v mericiach
- `area_achtel` – rozloha v achtloch
- `type` – označuje typ pola
- `class_1_merice` – rozloha pola 1. triedy v mericiach
- `class_1_achtel` – rozloha pola 1. triedy v achtloch
- `class_2_merice` – rozloha pola 2. triedy v mericiach
- `class_2_achtel` – rozloha pola 2. triedy v achtloch
- `class_3_merice` – rozloha pola 3. triedy v mericiach
- `class_3_achtel` – rozloha pola 3. triedy v achtloch

## 6.4 Poddanská přiznávací fase

„Poddanská přiznávací fase“, známe tiež ako „Rektifikační akta“, slúžil ako súpis pôdy poddaných na Morave[3]. Bude potrebné vytvoriť päť nových tabuliek pre prepis tohto historického prameňa. Schémy všetkých týchto tabuliek, spolu s už existujúcimi tabuľkami, je možné vidieť nižšie na obrázku 6.4.

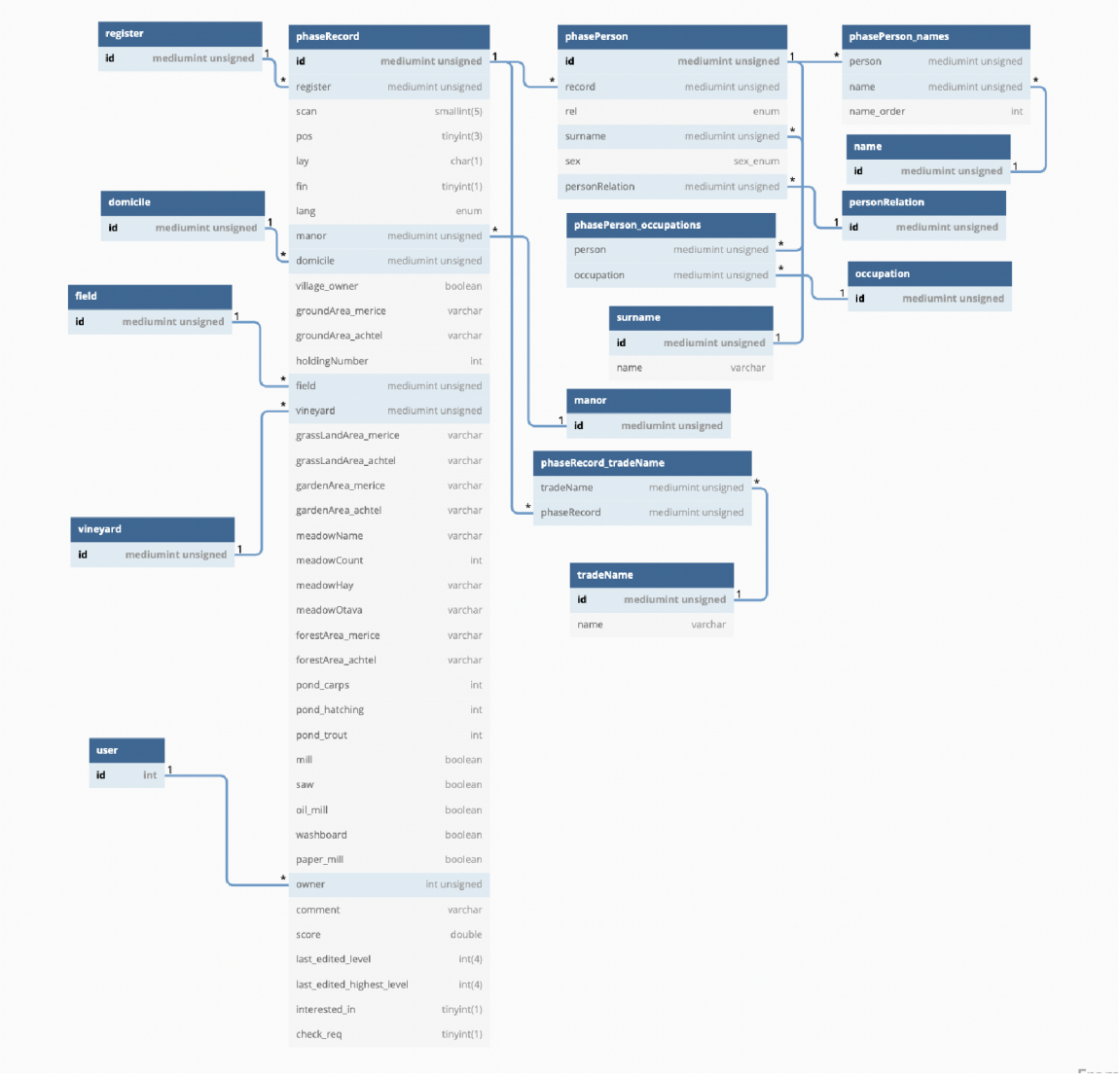
Tabuľka so záznamami o osobe, `phasePerson`, bude obsahovať len pár základných informácii ako priezvisko, pohlavie a odkaz na záznam. Rovnako ako pri predchádzajúcich prameňoch, tak aj tu si budeme viesť záznamy o osobách s rôznym vzťahom k záznamu. Jedná sa o osoby s označením:

- `phase_orig` – hlavná osoba spätá so záznamom
- `phase_orig_rel` – osoba, po ktorej bola hlavná osoba vdova/vdovec
- `phase_cur` – osoba, koho bola rodičom hlavnej osoby
- `phase_cur_rel` – osoba, koho bola rodičom hlavnej osoby

Podstatne viac dát bude obsahovať tabuľka o samotnom zázname.

Tabuľka o zázname, `phaseRecord`, bude obsahovať informácie ako je umiestnenie, rôzne výmery a informácie o viniciach, poliach, lúky, záhrady, .... Pre uchovávanie informácií o vinohradoch a poliach sa budú využívať už existujúce tabuľky, ktoré boli vytvorené pre zdroj „Lánové rejstříky“ doplnené o potrebné stĺpce (triedy polí).

Rovnako ako pri predchádzajúcich zdrojoch, tak aj tu sa budú viesť údaje spojené s osobou ako meno a zamestnanie, pre ktoré bude potrebné vytvoriť tabuľky s prepojeniami. Ostatné potrebné tabuľky, `name`, `occupation`, `personRelation`, `surname`, `user` a `register`, sa nebudú vytvárať, ale sa použijú už existujúce.




Obr. 6.4: Schémy tabuliek pre záznamy zo zdroja „Poddanská přiznávací fase“

### Popis niektorých stĺpcov tabuľky phasePerson

- `surname` – priezvisko osoby
- `sex` – pohlavie osoby
- `personRelation` – názov vzťahu k inej osobe

**Popis niektorých stĺpcov tabuľky**

- `manor` – názov panstva
- `domicile` – názov obce
- `village_owner` – označuje, či je držiteľom obec
- `groundArea_merice` – výmera lúk v mericiach
- `groundArea_achtel` – výmera lúk v achtloch
- `holdingNumber` – dražobné číslo
- `field` – informácie poli
- `vineyard` – informácie o vinohrade
- `grassLandArea_merice` – výmera pastvín v mericiach
- `grassLandArea_achtel` – výmera pastvín v achtloch
- `meadowName` – názov lúky
- `meadowCount` – počet lúk
- `meadowHay` – fúry sena
- `meadowOtava` – fúry otava
- `forestArea_merice` – výmera lesov v mericiach
- `forestArea_achtel` – výmera lesov v achtloch
- `pond_carps` – počet kaprov
- `pond_hatching` – počet násadových rýb
- `pond_trout` – počet pstruhov
- `mill` – označuje výskyt mlynu
- `saw` – označuje výskyt píli
- `oil_saw` – označuje výskyt olejárne
- `washboard` – označuje výskyt valchy
- `paper_mill` – označuje výskyt papierne

## 6.5 Berní rula

Posledným novým historickým zdrojom, ktorý sa bude integrovať do systému je „Berní rula“. Berní rula je historický prameň, ktorý slúžil ako súpis daňových povinností v Českom kráľovstve. Údaje, ktoré tento prameň obsahoval, boli využívané ako základ pre výber daní[4]. Pre ukladanie záznamov z tohto historického zdroja bude potrebné vytvoriť len tri nové tabuľky. Schémy všetkých týchto tabuliek, spolu s už existujúcimi tabuľkami, je možné vidieť nižšie na obrázku 6.5.

Pri výbere daní sa bralo do úvahy mnoho faktorov, ako napríklad počty rôznych druhov zvierat (ovce, jalovice, kravy, prasatá, atď.) alebo aj z počtu pastierov a ich pomocníkov a iné. Tieto informácie budu obsiahnuté v tabuľke so záznamom, `taxRecord`.

V tabuľke s osobou sa budú viesť len priezvisko, prípadne alternatívne priezvisko osoby. A ako pri predchádzajúcich osobách, tak aj tu sa budú zaznamenávať informácie o rôznych osobách:

- `tax_main` – hlavná osoba spätá so záznamom
- `tax_widow` – osoba, po ktorej bola hlavná osoba vdova/vdovec
- `tax_son_daughter` – osoba, koho bola rodičom hlavnej osoby

Posledná tabuľka, ktorú bude potrebné vytvoriť, je tabuľka ktorá bude mapovať osobu s jej menami. Na uchovanie ostatných potrebných informácií sa použijú už existujúce tabuľky, jedná sa o tabuľky `register`, `district`, `domicile`, `manor`, `surname`, `name` a `user`.

#### Popis niektorých stĺpcov tabuľky taxPerson

- `surname` – priezvisko osoby
- `alt_surname` – alternatívne priezvisko osoby
- `sex` – pohlavie osoby
- `mlst` – označuje ak existovala osoba s rovnakým menom, že bol mladší alebo starší
- `widow` – označuje, či osoba bola vdova/vdovec
- `son_daughter` – označuje, či osoba bola syn, či dcéra niekoho
- `roles` – počet rolí
- `saddledType` – označuje, či osoba bola sedliakom alebo chalúpnikom
- `other` – ostatné informácie

**Popis niektorých stĺpcov tabuľky taxRecord**

- `distrinct` – názov kraja
- `domicile` – názov obce
- `manor` – názov pastva
- `order` – poradie osadlých
- `sown_spring` – množstvo zasiatych obilnín na jar
- `sown_winter` – množstvo zasiatych obilnín na zimu
- `cows` – počet kráv
- `heifers` – počet jalovíc
- `sheeps` – počet oviec
- `pigs` – počet prasat
- `settled_1652` – počet novo usadlých v roku 1652
- `settled_1653` – počet novo usadlých v roku 1653
- `settled_1654` – počet novo usadlých v roku 1654
- `settled_1655` – počet novo usadlých v roku 1655
- `burns` – pohorelí
- `fasts` – pôsty
- `trades` – živnosti
- `shepherds_masters` – majstri pastieri
- `shepherds_grooms` – paholkovia
- `jews_over20` – počet židov nad 20 rokov
- `jews_10to20` – počet židov od 10 do 20 rokov


Obr. 6.5: Schémy tabuliek pre záznamy zo zdroja „Berní rula“

# Kapitola 7

# Implementácia

Nasledujúca podkapitola bude popisovať zapracovanie možnosti prepisu z ďalších historických prameňov do systému DEMoS. Pracoval som so zdrojovými súbormi napísanými v jazyku PHP, ktoré mi boli poskytnuté na platforme GitLab, hosťovanej na serveroch VUT[1]. Implementácia by mohla byť rozdelená do štyroch častí a to:

- Implementácia pohľadov, teda užívateľského rozhrania (kapitola 7.1)
- Implementácia prezentérov (kapitola 7.2)
- Implementácia tried, ktoré sú určené na prácu s jednotlivými historickými prameňmi (komunikácia s databázou, transformácia dát do vhodnej podoby, a podobne, kapitola 7.3)
- Ostatné úpravy (kapitola 7.4)

## 7.1 Pohľady

Pohľady tvoria časť aplikácie, ktorá má za úlohu zobrazenie informácií užívateľovi. V našom systéme tvorbe pohľadov pomáha spomínaný framework Latte 3.1. Samotné pohľady sú naimplementované v súboroch obsahujúce príponou `.latte`, ktoré môžeme nájsť v zložke na adrese `app/presenters/templates`. Výnimkou sú formuláre, ktoré sú naimplementované ako komponenty, ktoré sa nachádzajú na adrese `app/components`.

Nie všetky pohľady je potrebné naimplementovať od nuly. Predpripravené som mal formuláre pre jednotlivé historické zdroje, no aj tak som musel v nich vykonať pár zmien. Jedná sa o nastavenie informácií o registri pre daný záznam a informácie o autorovi prepisovaného záznamu, ktoré sa v daných pohľadoch automaticky nedoplňovali. Bolo potrebné pridať tieto informácie ako východiskovú hodnotu s využitím funkcie `setDefaultValue(...)` nad jednotlivými objektami vstupných polí v nasledujúcich triedach:

- `AddLandIndexFormControl`
- `AddPhaseFormControl`
- `AddTaxGneissFormControl`
- `AddUrbariumFormControl`

[1] http://perun.fit.vutbr.cz:4141

Avšak niektoré pohľady je potrebné vytvoriť, ako napríklad zoznam registrov, ktoré sú určené pre jednotlive historické zdroje (v databáze registrov sú označené stĺpcom "type") alebo náhľad konkrétneho registra a tabuľka s už existujúcimi prepísanými záznamami. Náhľad konkrétneho registra obsahuje informácie o registri, v ktorom sa používateľ aktuálne nachádza, ako napríklad jeho signatúra, jazyk, okres, v ktorom sa nachádza a iné.

V súvislosti s pohľadmi je dobré spomenúť, že v systéme je zapracovaný systém lokalizácie (momentálne je ale podporovaný len český jazyk), ktorý má pomáhať s prekladom systému do rôznych jazykov. Momentálne sa ale jednotlivé reťazce nachádzajú uložené v poli priamo v kóde aplikácie, čo nie je moc ideálne, no v budúcnosti je plán na presunutie týchto reťazcov do samostatných konfiguračných súborov. Tento zoznam všetkých reťazcov sa v čase písania tejto práce nachádza v súbore `MyTranslator.php`, nachádzajúci sa v zložke `app`. Tento súbor som rozšíril pridaním reťazcov, ktoré používam v nových pohľadoch ako hlavičky tabuliek a stránok.

## 7.2 Prezentéry

Prezentéry, ako už bolo spomenuté v kapitole 3.1, reaguje na požiadavky zo strany pohľadov a obsluhujú ich. Tieto triedy, ktoré reprezentujú prezentéry, sa nachádzajú v zložke na adrese `app/presenters`. Všetky vytvorené prezentéry v aplikácií dedia triedu `BasePresenter`, ktorá obsahuje inštancie tried potrebných pre komunikáciu s databázou.

Každá trieda obsahuje metódy, ktoré sa zavolajú pri načítaní určitej URL adresy, ktoré sú namapované na meno akcie. Namapovanie URL adresy na akcie bude popísané v texte nižšie. Názvy metód sa z pravidla začínajú slovom „render" nasledované názvom akcie. Napríklad názov metódy, ktorá bude zavolaná pre adresu „pridat-zaznam" namapovanej na akciu „addRecord" bude `renderAddRecord(...)`. Hlavnou úlohou týchto tried je naplnenie pohľadov potrebnými dátami z databázy.

Naimplementované boli nasledujúce prezentéry:

- `PhasePresenter` - prezentér pre historický zdroj „Poddanská přiznávací fase"
- `LandIndexPresenter` - prezentér pre historický zdroj „Lánové rejstříky"
- `TaxGneissPresenter` - prezentér pre historický zdroj „Berní rula"
- `UrbariumPresenter` - prezentér pre historický zdroj „Urbáře"

## 7.3 Manažéri

Manažérske triedy (často sa používa aj názov „Services", služby) sú triedy, ktoré v našom projekte slúžia na komunikáciu s databázou (vkladanie, získavanie záznamov) a na prácu s modelom dát pre daný historický zdroj. Manažérske triedy sa nachádzajú v zložke na adrese `app/model`. Naimplementované budú nasledujúce manažérske triedy:

- `PhaseManager` – manažérska trieda pre historický zdroj „Poddanská přiznávací fase"
- `LandIndexManager` – manažérska trieda pre historický zdroj „Lánové rejstříky"
- `TaxGneissManager` – manažérska trieda pre historický zdroj „Berní rula"
- `UrbariumManager` – manažérska trieda pre historický zdroj „Urbáře"

Prvou vecou, ktorú som vykonal bola úprava triedy `TableNames`. Túto triedu dedí trieda `BaseManage`, ktorá je základom pre každú manažérsku triedu. Trieda `TableNames` obsahuje informácie (názvy) o databázových tabuľkách, takže bolo potrebné tu zaevidovať názvy nových tabuliek, ktoré som vytvoril pre nové historické zdroje. Potrebné bolo aj pridanie týchto nových tried ako nové atribúty do triedy `BasePresenter`. Inicializácia týchto nových atribútov prebehne v konštruktore tejto triedy, kde o vytvorenie nových inštancií sa postará samotný Nette Framework.

Následne každá trieda, ktorú som naimplementoval dedí triedu `ScoreManager`, ktorá obsahuje potrebné metódy pre výpočet a uloženie skóre za prepísaný záznam. Novo naimplementované triedy obsahujú niekoľko spoločných metód, ktoré budú popísané v nasledujúcej časti.

### Metóda getModel($record, $edit)

Táto metóda vezme svoj parameter `$record`, ktorý obsahuje záznam z databázy a transformuje ho na model používaný v pohľadoch aplikácie. Transformované sú aj dáta z tabuliek, na ktoré sa odkazuje hlavná tabuľka prepisovaného záznamu s pomocou cudzieho kľúča a záznamy o osobách, ktoré sú v nejakom vzťahu so záznamom (stĺpec `rel` v tabuľke).

Dôvodom, prečo je potrebná táto metóda je, že názvy stĺpcov v databáze neodpovedajú identifikátorom jednotlivých polí vo formulároch. V aplikácií je zaužívaný formát `__názov_sekcie__názov_atribútu`. Druhým dôvodom je, že ak stĺpec v databáze nie je vyplnený, tak nie je ani vrátený pri požiadavke. Z tohoto dôvodu treba vytvoriť prázdne položky v modeli pre pohľady, inak riskujeme vyvolanie výnimky, že položka s daným názvom sa nenašla.

K dispozícii je aj voliteľný parameter `$edit` s predvolenou hodnotou `false`. Tento parameter slúži na určenie, či model je určený pre formuláre pre úpravu záznamu alebo pre zobrazenie daného záznamu v tabuľke prepísaných záznamov. Nutnosť tohto parametru je preto, lebo napríklad v tabuľke prepísaných záznamov niektorá sekcia, ktorá obsahuje atribút typu `enum` s implicitnou hodnotou (pri osobe to môže byť atribút s pohlavím nastavená na hodnotu „nešpecifikované“) a nebola vyplnená, chceme mať celú sekciu s prázdnymi reťazcami. Keby však boli tieto atribúty prázdne pri úprave záznamu, vrstva pohľadov by mala problém namapovať prázdnu hodnotu na hodnotu tohto atribútu, čo by vyvrcholilo vyvolaním výnimky. Z tohto dôvodu ak je nastavený parameter `$edit` na hodnotu `true`, tak namiesto prázdneho reťazca sa použije implicitná hodnota.

### Metóda getRecord($id)

Metóda `getRecord` slúži na získanie záznamu historického zdroja z databázy na základe jeho primárneho kľúča. Na prácu s databázou sa použije takzvaný „Nette Database Explorer“, ktorý nám umožní písať databázové požiadavky bez nutnosti písať SQL kód. Samotné získanie záznamu prebehne s pomocou príkazu 7.1.

```
$this->db->table("nazov tabulky")->get($id);
```

Výpis 7.1: Získanie záznamu z databázy na základe primárneho kľúča v PHP

### Metóda getByRegisterId($registerId, $limit, $offset)

Táto metóda slúži na získanie všetkých záznamov, ktoré sa nachádzajú v určitom registri a vráti ich ako pole modelov získaných s pomocou metódy `getModel`.

Táto metóda sa používa pri zobrazovaní všetkých prepísaných záznamov v danom registri. Tabuľka, ktorá zobrazuje tieto záznamy, zobrazuje len určitý počet prepísaných záznamov na jednej stránke. Ako východisková hodnota je nastavená na 25 záznamov na jednu stránku. Preto je potrebné z databázy získať len určitý rozsah záznamov. Toto môžeme docieliť tak, že metóda obsahuje parametre `$limit` a `$offset`. Parameter `$limit` obmedzuje maximálny počet záznamov, ktoré sa získajú z databázy. Druhý parameter `$offset` má za úlohu určiť koľko záznamov sa má preskočiť. Tento požadovaný efekt môžeme docieliť použitím metódy `limit` nad vrátenými záznamami. Použitie je možné vidieť v príkaze 7.2.

```
$records = $this->db->table($this->phaseRecordTable)
                    ->where('register = ?', $registerId)
                    ->limit($limit, $offset);
```

Výpis 7.2: Obmedzenie rozsahu získaných záznamov

### Metóda saveRecord($raw_data)

Následujúca metóda slúži na uloženie záznamu do databázy. Ako vstup dostane metóda pole hodnôt získaných z formulára na pridanie, prípadne na úpravu záznamu.

Pred začiatkom procesu ukladania sa prečistia vstupné dáta od položiek s prázdnymi reťazcami, prípadne s reťazcami, ktoré obsahujú len biele znaky. Na jednotlivé položky sa použije funkcia `trim`, ktorá zmaže všetky biele znaky na začiatku a konci daného reťazca. Následne sa skontroluje dĺžka reťazca a ak sa rovná nule, nastaví sa hodnota na hodnotu `null`. Tento krok sa vykonáva preto, aby sa zabránilo vytváraniu prázdnych záznamov v tabuľkách.

Následne sa vytvorí nová transakcia a začne sa mapovanie hodnôt z formulára na stĺpce z tabuľky. Prvými hodnotami sú tie, ktoré sa priamo nachádzajú v tabuľke so záznamom (napríklad jazyk, orientácia záznamu, ...). Ďalej sa budú mapovať dáta, na ktoré sa hlavný záznam odkazuje s pomocou cudzieho kľúča (napríklad tabuľka o vinohrade alebo o poli). Skontroluje sa, či je zadaná aspoň jedna hodnota vo formulári pre daný záznam, ak áno vytvorí sa nový záznam pre danú entitu, uloží sa a jej primárny kľúč, ktorý sa následne uloží v tabuľke prepisovaného záznamu ako cudzí kľúč.

Po uložení všetkých záznamov, na ktoré sa odkazuje hlavná tabuľka záznamu s pomocou cudzieho kľúča, nasleduje časť kde sa ukladá samotný záznam do tabuľky. Najprv sa skontroluje, či sa záznam upravuje alebo vytvára nový. Ak nieje uložený primárny kľúč záznamu v dátach z formulára (skryté pole s identifikátorom „___record___id“), tak sa jedná o vytvorenie nového záznamu. Potom sa teda nastaví aktuálne prihlásený používateľ, ktorý pridáva záznam, ako vlastník záznamu a uloží sa nový záznam do databázy. Inak sa aktualizuje stupeň oprávnení užívateľa pri poslednej úprave a najvyšší stupeň oprávnení a aktualizuje sa záznam v databáze.

Nasleduje časť ukladania informácií o osobe, ktorá je v nejakom vzťahu s prepísaným záznamom (vlastník pozemku, príbuzný vlastníka, dieťa vlastníka, ...). Tenko krok sa vykonáva až po uložení hlavného záznamu preto, lebo je potrebné vedieť primárny kľúč záznamu, ktorý sa nastaví v zázname osoby (stĺpec `record`). Podobne ako pri záznamoch vyššie, tak aj tu sa skontroluje, či bola zadaná aspoň jedna informácia o osobe. Ak nebola zadaná žiadna hodnota, ale je nastavený primárny kľúč osoby, tak to znamená, že informácie o osobe boli zmazané zo záznamu a preto sa prečistí tabuľka s jej informáciami. Ak bola zadaná nejaká hodnota, tak sa vytvorí alebo upraví záznam o osobe.

Poslednou časťou metódy je výpočet a uloženie skóre, ktoré je popísane nižšie a potvrdenie vytvorenej databázovej transakcie.

### Pomocné metódy

Posledným typom metód, ktoré sa nachádzajú v manažérskych triedach sú pomocné metódy. Sú to metódy, ktoré premazávajú záznam o osobe, upravujú jej mená (zmazanie mien, ktoré boli pri úprave vymazané alebo pridanie nových mien) alebo zamestnania.

Rozšírená bola aj trieda `BaseManager`, kde boli pridané metódy, ktoré by boli identické pre každú manažérsku triedu. Pridané boli metódy určené k získaniu priezviska, vzťahu a zázemia z databázy na základe ich primárneho kľúča a metóda, ktorá na základe registra a názvu tabuľky prepisovaného záznamu vráti počet záznamov z daného registra.

### Výpočet skóre záznamu

Výpočet skóre za prepísaný záznam prebieha v triede `ScoreManager` volaním metódy pre príslušný historický prameň. Jedná sa o metódy:

- `getUrbariumRecordScore` – výpočet skóre pre historický zdroj „Urbáře“
- `getPhaseRecordScore` – výpočet skóre pre historický zdroj „Poddanská přiznávací fase“
- `getTaxGneissRecordScore` – výpočet skóre pre historický zdroj „Berní rula“
- `getLandIndexRecordScore` – výpočet skóre pre historický zdroj „Lánové rejstříky“

Všetky tieto metódy obsahujú parameter, ktorým je primárny kľúč záznamu. Výsledné skóre, ktoré užívateľ dostane, ovplyvňuje niekoľko faktorov, a to jazyk, v ktorom bol originálny záznam a či je záznam kompletne prepísaný. Skóre sa vypočíta tak, že sa vytvorí databázový požiadavok, ktorý spočíta všetky neprázdne stĺpce záznamu a toto číslo sa vynásobí konštantou, ktorá je daná jazykom prepisovaného záznamu. Tento výpočet sa vykoná aj pre záznamy z tabuliek, na ktoré sa daný záznam odkazuje s pomocou cudzieho kľúča a pre záznamy z tabuliek osôb, ktoré sú vo vzťahu s daným záznamom. Tieto metódy sa volajú na konci metódy `saveRecord` pred potvrdením databázovej transakcie.

## 7.4 Ostatné

Mimo spomínané naimplementované triedy je potrebný aj zásah do konfiguračných súborov. Prvá vec, ktorú bolo potrebné vykonať je registrácia služieb (v našom prípade tried označených ako “manažér”). Registrácia služieb prebieha v súbore `config.neon`, nachádzajúci sa v priečinku `/app/config`. Dôvodom prečo je potrebné zaregistrovať službu je, že Nette framework využíva princíp zvaný „Dependecy Injection“, ktorý pomáha pri vkladaní závislostí medzi jednotlivými objektami v programe. Z tohto dôvodu aby framework vedel nájsť nájsť dané triedy, tak je potrebné ich niekde zaregistrovať. Toto sa vykonáva v spomínanom konfiguračnom súbore v časti „services:“, kde sa zadá odkaz na danú triedu.

Druhou vecou pre správne fungovanie novo naimplementovaných častí je namapovanie url adries k akciám v jednotlivých prezenteroch. Toto mapovanie sa vykonáva v súbore `RouterFactory.php`, nachádzajúcom sa v zložke `app/router`.

# Kapitola 8

# Testovanie

Jedným z posledných bodov tejto práce je testovanie aplikácie. Testovanie je dôležitou súčasťou vývojového cyklu aplikácie, ktoré má za úlohu objaviť chyby vzniknuté počas vývoja a návrhu aplikácie. Testovanie novo vykonaných zmien systému by som rozdelil na dve časti, a to testovanie užívateľského rozhrania a testovanie funkčnosti, čím sa sa bude zaoberať následujúca kapitola.

## 8.1 Užívateľské testovanie

Užívateľské testovanie je jedna z najpoužívanejších metód testovania užívateľského rozhrania, kde sa za počítač posadí používateľ, ktorý používa danú aplikáciu. Tomuto používateľovi sú zadané úlohy, ktoré má vykonať v danom systéme a je popri používaní aplikácie pozorovaný. Takýto typ testovania môže odhaliť rôzne chyby a nedostatky užívateľského rozhrania, ktoré samotný vývojári mohli prehliadnuť tým, že ich nepovažovali za chyby. Môže sa jednať napríklad o chyby funkčnosti, prípadne sa môže užívateľské rozhranie zdať používateľovi neprehľadné. Pohľad používateľa na systém môže byť iný ako vývojára, ktorý dané rozhranie naimplementoval a pozná ako funguje, čo môže výrazne pomôcť k vylepšeniu používateľnosti aplikácie.

Tento typ testovania by bolo vhodné vykonať, keby sa táto práca zaoberala návrhom a implementáciou užívateľského rozhrania. Táto práca mala za cieľ integrovať nové historické zdroje do už existujúceho systému a užívateľského rozhrania. Keď som začínal s touto prácou, tak v pláne bolo aj návrh a implementácia niektorých častí užívateľského rozhrania, no neskôr sa zistilo, že skoro všetko už bolo naimplementované a navrhnuté. Síce bola potrebná implementácia niektorých pohľadov, no tie vychádzali z už vytvoreného návrhu a jednalo sa len o komponenty obsahujúce zoznamy položiek, prípadne tabuľka s informáciami. Keďže nebolo hlavnou úlohou tejto práce návrh a implementácia užívateľského rozhrania, tak užívateľské testovanie sa nebude vykonávať, keďže momentálne nedáva moc zmysel.

## 8.2 Chýbajúce informácie o registroch

Pred začatím testovania som zistil, že v databáze chýbajú niektoré informácie o registroch. Chýba tu informácia o tom, kde sa daný register nachádza. Presnejšie sa jednalo o stĺpec `admin_id`, ktorý určuje, ktorá administračná jednotka spravuje daný register. Druhý chýbajúci údaj bol záznam v tabuľke `register_municip` určujúci v akej samospráve sa daný

register nachádza. Dôvodom je, že na rozdiel od matrík, ktoré sa nachádzali v určitej obci a pokrývali určitú oblasť, tak v mnou integrovaných historických prameňoch to tak nieje.

## 8.3 Testovanie novo vykonaných zmien

V systéme som po pridaní nový zdrojov testoval pridávanie a úpravu nových záznamov. Účelom tohto testovania bolo odhalenie chýb komunikácie medzi pohľadom a triedami zabezpečujúcimi komunikáciu s databázou. Súčasťou testovaných scenárov bola aj kontrola výpočtu skóre, ktoré dostal používateľ za pridanie alebo úpravu záznamu. Pri vyplňovaní polí, ktoré môžu nadobúdať viacero hodnôt (napríklad meno osoby alebo zamestnanie) som skúšal zadať rôzny počet hodnôt. Pre vytváranie nových záznamov som otestoval nasledujúce scenáre prepisov:

1. nový záznam len s povinnými údajmi (poradie skenu, poradie záznamu, ...)
2. nový záznam so všetkými vyplnenými poliami
3. nový záznam s čiastočne vyplnenými poliami

Na prvý scenár sa použije záznam, ktorý obsahuje len základné údaje o danom zázname. Jedná sa o „Pořadí scanu“ a „Pořadí záznamu“. Pri urbároch sú navyše vyplnené „Pagina“, „Folio“ a „Rok sepsání“. V druhom a treťom scenári som vyplnil polia formuláru s hodnotami, ktorých príklad sa nachádzaj v prílohe A. Celkovo som vykonal 5 prepisov pre každý historický prameň a pre každý scenár, teda celkom 60 prepisov.

Úprava záznamov využíva rovnaký komponent, ako formulár na pridávanie nových záznamov. Jediným rozdielom je len, že pri úprave sa nastavujú hodnoty polí vo formulároch informáciami o zázname, ale aj tak som otestoval aj úpravu záznamov. Pri testovaní úpravy záznamov som používal tie, ktoré som vytvoril pri testovaní prepisu nových záznamov. Skúšal som nasledujúce scenáre úprav:

1. úprava prázdneho záznamu vyplnením všetkých polí
2. zmazanie všetkých nepovinných polí pri plne vyplnenom zázname
3. úprava plne vyplneného záznamu

Pri prvom scenári som zase len upravil povinné polia formulárov pre jednotlivé zdroje, ktoré sú popísané vyššie pri testovaní vytvárania nových záznamov a rovnako na druhý aj tretí scenár som použil hodnoty, ktorých príklad sa nachádzaj v prílohe A.

Testovanie ukázalo problém, že chýbajú informácie o registroch, ktoré nie som schopný opraviť. Ostatné chyby ktoré súviseli s novo naimplementovanými triedami boli opravené (prevažne preklepy v identifikátoroch).

# Kapitola 9

# Záver

Táto práca mala dva ciele. Prvým cieľom bola analýza databázových systémov a aktuálne používaného databázového systému. Účelom tejto analýzy bolo určiť, či bol daný systém (v našom prípade MySQL) vhodne zvolený, teda či spĺňa požadované vlastnosti. Preto bolo potrebné sa aj pozrieť na iné databázové systémy. Analyzované boli ďalšie relačné databázy, presnejšie Oracle Database a PostreSQL. Mimo relačných som sa pozrel aj na nerelačné, a to na grafové databázy so zástupcom Neo4j a na dokumentové databázy so zástupcom MongoDB. Z vykonanej analýzy som dospel k tomu, že zmena by priniesla zlepšenie, ale cena za túto zmenu by bola väčšia ako samotné prínosy. Významné zlepšenie by bolo pri použití grafových databáz na ukladanie vzťahov medzi osobami, ktoré by mohli byť použité v rodokmeňoch, alebo iných modeloch. Pri ukladaní ostatných dát systému (užívatelia systému, štáty, okresy, a podobne) a samotných prepísaných záznamov by to nemalo až taký zmysel.

Súčasťou analýzy bola aj revízia aktuálnych databázových schém, kde som vykonal zmeny typov niektorých stĺpcov. Hlavne sa jednalo o zmenu typ `TEXT` na `VARCHAR`. Druhov zmenou bolo zavedenie indexov pre stĺpce, ktoré sa aktuálne používajú pri vyhľadávaní (stĺpec `signature` v tabuľke `registry`) a stĺpcov, ktoré by potenciálne mohli byť využité v budúcnosti ako základom pre vyhľadávanie.

Hlavným cieľom tejto diplomovej práce bolo integrovať ďalšie historické zdroje do systému DEMoS, ktorý je postavený na frameworku Nette. Integrované boli všetky po mne požadované historické pramene. Jedná sa o „Berní rule“, „Urbáre“, „Lánové rejstříky“ a „Poddanská přiznávací fase“. Pri integrácií boli navrhnuté nové tabuľky pre tieto pramene. Naimplementované boli potrebné pohľady a triedy, ktoré sa starali o komunikáciu s týmito novými tabuľkami a ktoré sa používali na manipuláciu s daným dátovým modelom.

Projekt DEMoS je pomerne veľký projekt, na ktorom je ešte veľa práce. V dobe písania tejto práce sa pracovalo na systéme, ktorý by zaznamenával zmeny vykonávané v jednotlivých záznamoch, ktorý nebol naimplementovaný pre mnou integrované pramene. Ďalšou vecou, ktorá sa môže stať ako predmet ďalších prác môže byť integrácia ďalších historických prameňov. Jedná sa o „Pozemkové knihy“, „Sčítací operáty“ a „Soupis poddaných dle víry“. Potrebná je aj úprava informácií o registroch, kde chýbajú informácie o administračných jednotkách a umiestnenie niektorých registrov, alebo úprava užívateľského rozhrania, ktoré by dovolovalo vyhľadávať tieto nové registre.

# Literatúra


[1] Borůvka, F. *GENEALOGIE v praxi. 3. přednáška Badatelské postupy, archívy, matriky a jejich zpracování* [online]. 2017 [cit. 2021-12-17]. Dostupné z: https://docplayer.cz/105663911-Genealogie-v-praxi-3-prednaska-badatelske-postupy-archivy-matriky-a-jejich-zpracovani.html.

[2] Borůvka, F. *GENEALOGIE v praxi. 9. přednáška Formy genealogických tabulek, možnosti zpracování* [online]. 2017 [cit. 2021-12-17]. Dostupné z: https://docplayer.cz/42656387-Genealogie-v-praxi-9-prednaska-formy-genealogickych-tabulek-moznosti-zpracovani.html.

[3] Drápela M., Podhrázský Z., Stachoň Z., Tajovská K.. *TEREZIÁNSKÝ KATASTR - TŘETÍ A ČTVRTÁ BERNÍ RULA* [online]. [cit. 2022-3-24]. Dostupné z: https://web.archive.org/web/20080316094058/http://www.geogr.muni.cz/ucebnice/dejiny/obsah.php?show=103.

[4] Drápela M., Podhrázský Z., Stachoň Z., Tajovská K.. *TEREZIÁNSKÝ KATASTR - TŘETÍ A ČTVRTÁ BERNÍ RULA* [online]. [cit. 2022-3-22]. Dostupné z: https://web.archive.org/web/20080316113159/http://www.geogr.muni.cz/ucebnice/dejiny/obsah.php?show=1013.

[5] Gilbert, S. a Lynch, N. Perspectives on the CAP Theorem. *Computer*. 2012, zv. 45, č. 2, s. 30–36. DOI: 10.1109/MC.2011.389.

[6] Greenwald, Rick and Stackowiak, Robert and Stern, Jonathan. *Oracle essentials: Oracle database 12c*. Ö'Reilly Media, Inc.", 2013. ISBN 978-1449343033.

[7] Group, T. P. *What can PHP do?* [online]. [cit. 2021-11-25]. Dostupné z: https://www.php.net/manual/en/intro-whatcando.php.

[8] Jaroslav Zendulka, V. B. *Relační model dat* [online]. 2020 [cit. 2021-11-25]. Dostupné z: https://wis.fit.vutbr.cz/FIT/st/cfs.php?file=%2Fcourse%2FIDS-IT%2Flectures%2Fcz%2F3_relmod.pdf.

[9] Lerdorf, R., Tatroe, K., Kaehms, B. a McGredy, R. *Programming Php*. Ö'Reilly Media, Inc.", 2002. ISBN 1-56592-610-2.

[10] MongoDB Inc.. *MongoDB Licensing* [online]. [cit. 2021-11-19]. Dostupné z: https://www.mongodb.com/community/licensing.

[11] MongoDB Inc.. *Why Use MongoDB and When to Use It?* [online]. [cit. 2021-10-20]. Dostupné z: https://www.mongodb.com/why-use-mongodb.

[12] MyHeritage Ltd.. *About MyHeritage* [online]. [cit. 2021-12-26]. Dostupné z: https://www.myheritage.com/about-myheritage/.

[13] MyHeritage Ltd.. *Subscription pricing* [online]. [cit. 2021-12-26]. Dostupné z: https://www.myheritage.com/pricing.

[14] Neo4j, Inc. *Native Graph Database* [online]. [cit. 2021-11-22]. Dostupné z: https://neo4j.com/product/neo4j-graph-database/.

[15] Neo4j, Inc. *Neo4j Licensing* [online]. [cit. 2021-11-22]. Dostupné z: https://neo4j.com/licensing/.

[16] Nette Foundation. *Licenční politika* [online]. [cit. 2021-11-25]. Dostupné z: https://nette.org/cs/license.

[17] Nette Foundation. *Začínáme s Latte* [online]. [cit. 2021-11-25]. Dostupné z: https://latte.nette.org/cs/guide.

[18] Nette Foundation. *Začínáme s Tracy* [online]. [cit. 2021-11-25]. Dostupné z: https://tracy.nette.org/cs/guide.

[19] Oracle Corporation. *11.3.4 The BLOB and TEXT Types* [online]. [cit. 2021-12-9]. Dostupné z: https://dev.mysql.com/doc/refman/8.0/en/blob.html.

[20] Oracle Corporation. *8.3.1 How MySQL Uses Indexes* [online]. [cit. 2021-12-02]. Dostupné z: https://dev.mysql.com/doc/refman/8.0/en/mysql-indexes.html.

[21] Oracle Corporation. *Chapter 11 Data Types* [online]. [cit. 2021-11-20]. Dostupné z: https://dev.mysql.com/doc/refman/8.0/en/data-types.html.

[22] Oracle Corporation. *Chapter 16 Alternative Storage Engines* [online]. [cit. 2021-11-20]. Dostupné z: https://dev.mysql.com/doc/refman/8.0/en/storage-engines.html.

[23] Oracle Corporation. *Database Licensing Information User Manual* [online]. [cit. 2021-10-15]. Dostupné z: https://docs.oracle.com/database/121/DBLIC/editions.htm#DBLIC109.

[24] Oracle Corporation. *The Main Features of MySQ* [online]. [cit. 2021-11-20]. Dostupné z: https://dev.mysql.com/doc/refman/8.0/en/features.html.

[25] Oracle Corporation. *MySQL Products* [online]. [cit. 2021-10-24]. Dostupné z: https://www.mysql.com/products/.

[26] Oracle Corporation. *Oracle Database XE Frequently Asked Questions* [online]. [cit. 2021-10-15]. Dostupné z: https://www.oracle.com/cz/database/technologies/appdev/xe/faq.html.

[27] Pine, L. G. *Genealogy* [online]. [cit. 2021-12-07]. Dostupné z: https://www.britannica.com/topic/genealogy.

[28] Quigley, E. a Gargenta, M. *PHP and MySQL by Example*. Pearson Education, 2006. ISBN 9780138006020. Dostupné z: https://books.google.cz/books?id=mPzUH1EPEJkC.

[29] Reif, J. How Do You Know If a Graph Database Solves the Problem? [online]. November 2018, [cit. 2021-10-19]. Dostupné z: https://neo4j.com/developer-blog/how-do-you-know-if-a-graph-database-solves-the-problem/.

[30] Skřebská, Z. *Genealogie a tvorba rodokmenů*. Brno, CZ, 2017. Bakalárska práca. Masarykova univerzita, Přírodovědecká fakulta. Dostupné z: https://is.muni.cz/th/mrrs2/Bakalarska_prace_-_Skrebska.pdf.

[31] SLOVNIKCUDZICHSLOV.SK. *Slovník cudzích slov* [online]. [cit. 2021-12-17]. Dostupné z: http://slovnikcudzichslov.sk/slovo/proband,%20probant.

[32] solid IT gmbh. *Method of calculating the scores of the DB-Engines Ranking* [online]. [cit. 2021-10-14]. Dostupné z: https://db-engines.com/en/ranking_definition.

[33] The PostgreSQL Global Development Group. *About* [online]. [cit. 2021-11-1]. Dostupné z: https://www.postgresql.org/about/.

[34] The PostgreSQL Global Development Group. *License* [online]. [cit. 2021-11-1]. Dostupné z: https://www.postgresql.org/about/licence/.

[35] Ullman, J. *Object-Relational Features of Oracle* [online]. [cit. 2021-10-15]. Dostupné z: http://infolab.stanford.edu/~ullman/fcdb/oracle/or-objects.html.

[36] Urbár Liptovská Kokava. *VZNIK URBÁROV* [online]. [cit. 2022-3-24]. Dostupné z: https://www.urbarkokava.sk/historia/vznik-urbarov/.

[37] Vaish, G. *Getting started with NoSQL*. 1. vyd. Packt Publishing, 2013. ISBN 978-1-84969-4-988.

[38] Vukotic, A., Watt, N., Abedrabbo, T., Fox, D. a Partner, J. *Neo4j in action*. Manning Shelter Island, 2015. ISBN 978-1-61729-0-763.

[39] Vysoké učení technické v Brne, Masarykova univerzita. *O projektu DEMoS* [online]. [cit. 2021-11-19]. Dostupné z: http://perun.fit.vutbr.cz/napoveda/o-projektu/.

[40] Wikipedie. *Matrika — Wikipedie: Otevřená encyklopedie* [online]. 2021 [cit. 2021-12-7]. Dostupné z: https://cs.wikipedia.org/w/index.php?title=Matrika&oldid=20172091.

[41] Wikipedie. *Lánový rejstřík* [online]. 2022 [cit. 2022-3-23]. Dostupné z: https://cs.wikipedia.org/wiki/L%C3%A1nov%C3%BD_rejst%C5%99%C3%ADk.

[42] Zhang, Y. a Luo, Y. An architecture and implement model for Model-View-Presenter pattern. In: *2010 3rd International Conference on Computer Science and Information Technology*. 2010, sv. 8, s. 532–536. DOI: 10.1109/ICCSIT.2010.5565090.

# Príloha A

# Testovacie záznamy na prepis

## A.1 Nové záznamy z Berní rule

### A.1.1 Plne vyplnený formulár

Pozice:

- Pořadí scanu – 1
- Rozložení na scanu – Vpravo
- Pořadí záznamu – 1
- Jazyk – Staročeský

Místo:

- Kraj – Kladno
- Panství – Panstvo 1
- Obec – Blevice

Osedlý:

- Sedlák/Chalupník – Sedlák
- Jméno – Honza
- Příjmení – Chaloupek
- Alternatívni příjmení – Chaloupka
- Ml./St. – áno
- Jiný – ine
- Pohlaví – Muž
- Vdovec/Vdova – áno
- Syn/Dcéra – áno
- Rolí – 5

Vdovec/Vdova po:

- Jméno – Marie
- Příjmení – Chaloupková

Syn/Dcéra koho:

- Jméno – Honza
- Příjmení – Chaloupek

Oseto:

- Na zimu – 4
- Na jaro – 1

Nově osedlý:

- 1652 – 7
- 1653 – 18
- 1654 – 11
- 1655 – 15
- Pohořelí – 2
- Půstky – 7
- Živnosti – 5
- Potahy – 1

Dobytek:

- Krávy – 9
- Jalovice – 5

- Ovce – 31
- Svině – 15
- Fara – 1

Ovčáci:

- Mistři – 2
- Pacholek – 5

Židé:

- Nad 20 let – 18
- 10-20 let – 24

Poznámka:

- Poznámka – Poznámka

### A.1.2 Čiastočne vyplnený formulár

Pozice:

- Pořadí scanu – 8
- Rozložení na scanu – Vpravo
- Pořadí záznamu – 1
- Jazyk – Neznámé

Místo:

- Kraj –
- Panství –
- Obec –

Osedlý:

- Sedlák/Chalupník – Sedlák
- Jméno – A??rej
- Příjmení – Novák
- Alternatívni příjmení –
- Ml./St. – nie
- Jiný –
- Pohlaví – Muž
- Vdovec/Vdova – nie
- Syn/Dcéra – nie
- Rolí –

Vdovec/Vdova po:

- Jméno –
- Příjmení –

Syn/Dcéra koho:

- Jméno –
- Příjmení –

Oseto:

- Na zimu –
- Na jaro –

Nově osedlý:

- 1652 – 11
- 1653 –
- 1654 –
- 1655 –
- Pohořelí – 2
- Půstky –
- Živnosti –
- Potahy –

Dobytek:

- Krávy –
- Jalovice – 25
- Ovce –
- Svině – 5
- Fara –

Ovčáci:

- Mistři –
- Pacholek –

Židé:

- Nad 20 let –
- 10-20 let –

Poznámka:

- Poznámka –

## A.2 Úprava záznamu z Berní rule

### A.2.1 Plne vyplnený formulár

Pozice:

- Pořadí scanu – 1
- Rozložení na scanu – Vpravo
- Pořadí záznamu – 1
- Jazyk – Staročeský

Místo:

- Kraj – Kladno →
- Panství – Panstvo 1 → Panstvo 2
- Obec – Blevice →

Osedlý:

- Sedlák/Chalupník – Sedlák → Chalupík
- Jméno – Honza → Ján
- Příjmení – Chaloupek → Caloupek
- Alternatívni příjmení – Chaloupka → Caulopka
- Ml./St. – áno → nie
- Jiný – ine
- Pohlaví – Muž
- Vdovec/Vdova – áno
- Syn/Dcéra – áno
- Rolí – 5 → 2

Vdovec/Vdova po:

- Jméno – Marie → Jana
- Příjmení – Chaloupková → Caloupková

Syn/Dcéra koho:

- Jméno – Honza → Albert
- Příjmení – Chaloupek → Caloupek

Oseto:

- Na zimu – 4 → 7
- Na jaro – 1 → 4

Nově osedlý:

- 1652 – 7 → 11
- 1653 – 18 → 15
- 1654 – 11 → 10
- 1655 – 15 → 21
- Pohořelí – 2 → 1
- Půstky – 7 → 6
- Živnosti – 5 → 4
- Potahy – 1 →

Dobytek:

- Krávy – 9 → 7
- Jalovice – 5 → 11
- Ovce – 31 → 29
- Svině – 15 →
- Fara – 1 →

Ovčáci:

- Mistři – 2 → 1
- Pacholek – 5 → 6

Židé:

- Nad 20 let – 18 → 19
- 10-20 let – 24 → 26

Poznámka:

- Poznámka – Poznámka → Poznámka 2

### A.2.2 Čiastočne vyplnený formulár

Pozice:

- Pořadí scanu – 8
- Rozložení na scanu – Vpravo
- Pořadí záznamu – 1
- Jazyk – Neznámé

Místo:

- Kraj –
- Panství – → Panstvo 2
- Obec –

Osedlý:

- Sedlák/Chalupník – Sedlák
- Jméno – A??rej
- Příjmení – Novák
- Alternatívni příjmení –
- Ml./St. – nie
- Jiný –
- Pohlaví – Muž
- Vdovec/Vdova – nie → áno
- Syn/Dcéra – nie
- Rolí –

Vdovec/Vdova po:

- Jméno – → Anna
- Příjmení – → Nováková

Syn/Dcéra koho:

- Jméno –
- Příjmení –

Oseto:

- Na zimu –
- Na jaro –

Nově osedlý:

- 1652 – 11
- 1653 – → 12
- 1654 –
- 1655 –
- Pohořelí – 2
- Půstky –
- Živnosti –
- Potahy –

Dobytek:

- Krávy –
- Jalovice – 25
- Ovce –
- Svině – 5 → 6
- Fara –

Ovčáci:

- Mistři –
- Pacholek –

Židé:

- Nad 20 let –
- 10-20 let –

Poznámka:

- Poznámka – → Poznámka

## A.3 Nové záznamy z Urbáře

### A.3.1 Plne vyplnený formulár

Pozice:

- Folio 1
- Pagina 1
- Pořadí scanu – 1
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Umístění:

- Rok sepsání – 1888
- Lokalita – Hobšovice
- Pořadí gruntu na scanu – 3
- Název gruntu – Názov 1

Původní držitel:

- Grunt původně pustý/spálený – áno
- Jméno – Jozef
- Příjmení – Majka
- Povolání – Pilčík
- Číslo popisné – 123
- Žid – áno
- Zařazení – Zaradenie 1
- Vztah k jiné osobě – sused
- Jméno této osoby – Robert
- Příjmení této osoby – Novák
- Penežní dávky – áno
- Naturálni dávky – áno
- Roboty – áno
- Režijní podnikání, další majetky a práva – áno

Následující držitel 1:

- Grunt původně pustý/spálený – áno
- Jméno – Alfonz
- Příjmení – Zápalka
- Povolání – Hrnčiar
- Číslo popisné – 333
- Zařazení – Zaradenie 2
- Vztah k jiné osobě – kamarát
- Jméno této osoby – Marek
- Příjmení této osoby – Mak
- Rok nabytí gruntu – 1890
- Penežní dávky – áno
- Naturálni dávky – áno
- Roboty – áno
- Režijní podnikání, další majetky a práva – áno

Následující držitel 2:

- Grunt původně pustý/spálený – áno
- Jméno – Martin
- Příjmení – Kováč
- Povolání – úradník
- Číslo popisné – 358
- Zařazení – Zaradenie 3
- Vztah k jiné osobě – syn
- Jméno této osoby – Filip
- Příjmení této osoby – Kováč
- Rok nabytí gruntu – 1893
- Penežní dávky – áno
- Naturálni dávky – áno
- Roboty – áno

- Režijní podnikání, další majetky a práva – áno

Následující držitel 3:

- Grunt původně pustý/spálený – áno
- Jméno – Martina
- Příjmení – Michalko
- Povolání – krajčírka
- Číslo popisné – 399
- Zařazení – Zaradenie 4
- Vztah k jiné osobě – manželka
- Jméno této osoby – Ján Jakub
- Příjmení této osoby – Michalko
- Rok nabytí gruntu – 1899
- Penežní dávky – áno
- Naturálni dávky – áno
- Roboty – áno
- Režijní podnikání, další majetky a práva – áno

Poznámka:

- Poznámka – Poznámka

### A.3.2 Čiastočne vyplnený formulár

Pozice:

- Folio 2
- Pagina 1
- Pořadí scanu – 4
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Umístění:

- Rok sepsání – 1898
- Lokalita –
- Pořadí gruntu na scanu – 8
- Název gruntu – Názov 2

Původní držitel:

- Grunt původně pustý/spálený –
- Jméno – Karol
- Příjmení –
- Povolání –
- Číslo popisné – 436
- Žid –
- Zařazení –
- Vztah k jiné osobě –
- Jméno této osoby –
- Příjmení této osoby –
- Penežní dávky –
- Naturálni dávky – áno
- Roboty –
- Režijní podnikání, další majetky a práva –

Následující držitel 1:

- Grunt původně pustý/spálený –
- Jméno – Matúš
- Příjmení – Horný
- Povolání – Lesník
- Číslo popisné – 463
- Zařazení –
- Vztah k jiné osobě – kolega
- Jméno této osoby – Andrej
- Příjmení této osoby –
- Rok nabytí gruntu – 1899
- Penežní dávky –

- Naturálni dávky –
- Roboty –
- Režijní podnikání, další majetky a práva –

Následující držitel 2:

- Grunt původně pustý/spálený –
- Jméno –
- Příjmení –
- Povolání –
- Číslo popisné –
- Zařazení –
- Vztah k jiné osobě –
- Jméno této osoby –
- Příjmení této osoby –
- Rok nabytí gruntu –
- Penežní dávky –
- Naturálni dávky –
- Roboty –
- Režijní podnikání, další majetky a práva –

Následující držitel 3:

- Grunt původně pustý/spálený –
- Jméno –
- Příjmení –
- Povolání –
- Číslo popisné –
- Zařazení –
- Vztah k jiné osobě –
- Jméno této osoby –
- Příjmení této osoby –
- Rok nabytí gruntu –
- Penežní dávky –
- Naturálni dávky –
- Roboty –
- Režijní podnikání, další majetky a práva –

Poznámka:

- Poznámka – Poznámka

## A.4 Úprava záznamov z Urbáře

### A.4.1 Plne vyplnený formulár

Pozice:

- Folio 1
- Pagina 1
- Pořadí scanu – 1
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Umístění:

- Rok sepsání – 1888
- Lokalita – Hobšovice
- Pořadí gruntu na scanu – 3
- Název gruntu – Názov 1 → Názov 3

Původní držitel:

- Grunt původně pustý/spálený – áno → nie
- Jméno – Jozef
- Příjmení – Majka
- Povolání – Pilčík → Kominár
- Číslo popisné – 123
- Žid – áno → nie
- Zařazení – Zaradenie 1

- Vztah k jiné osobě – sused → kolega
- Jméno této osoby – Robert
- Příjmení této osoby – Novák
- Penežní dávky – áno → nie
- Naturálni dávky – áno → nie
- Roboty – áno → nie
- Režijní podnikání, další majetky a práva – áno → nie

Následující držitel 1:

- Grunt původně pustý/spálený – áno
- Jméno – Alfonz
- Příjmení – Zápalka
- Povolání – Hrnčiar
- Číslo popisné – 333 → 334
- Zařazení – Zaradenie 2 → Zaradenie 123
- Vztah k jiné osobě – kamarát
- Jméno této osoby – Marek
- Příjmení této osoby – Mak → Mako
- Rok nabytí gruntu – 1890 → 1891
- Penežní dávky – áno
- Naturálni dávky – áno → nie
- Roboty – áno → nie
- Režijní podnikání, další majetky a práva – áno → nie

Následující držitel 2:

- Grunt původně pustý/spálený – áno → nie
- Jméno – Martin → Matej
- Příjmení – Kováč
- Povolání – úradník
- Číslo popisné – 358
- Zařazení – Zaradenie 3
- Vztah k jiné osobě – syn
- Jméno této osoby – Filip
- Příjmení této osoby – Kováč
- Rok nabytí gruntu – 1893
- Penežní dávky – áno
- Naturálni dávky – áno
- Roboty – áno → nie
- Režijní podnikání, další majetky a práva – áno

Následující držitel 3:

- Grunt původně pustý/spálený – áno
- Jméno – Martina
- Příjmení – Michalko
- Povolání – krajčírka
- Číslo popisné – 399
- Zařazení – Zaradenie 4
- Vztah k jiné osobě – manželka
- Jméno této osoby – Ján Jakub
- Příjmení této osoby – Michalko
- Rok nabytí gruntu – 1899
- Penežní dávky – áno
- Naturálni dávky – áno
- Roboty – áno
- Režijní podnikání, další majetky a práva – áno

Poznámka:

- Poznámka – Poznámka

### A.4.2 Čiastočne vyplnený formulár

Pozice:

- Folio 2
- Pagina 1
- Pořadí scanu – 4
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Umístění:

- Rok sepsání – 1898
- Lokalita –
- Pořadí gruntu na scanu – 8
- Název gruntu – Názov 2

Původní držitel:

- Grunt původně pustý/spálený –
- Jméno – Karol
- Příjmení – → Vrána
- Povolání – → Starosta
- Číslo popisné – 436
- Žid –
- Zařazení –
- Vztah k jiné osobě – →
- Jméno této osoby –
- Příjmení této osoby –
- Penežní dávky – → áno
- Naturálni dávky – áno
- Roboty –
- Režijní podnikání, další majetky a práva – → áno

Následující držitel 1:

- Grunt původně pustý/spálený – → áno
- Jméno – Matúš → Tomáš
- Příjmení – Horný → Dolný
- Povolání – Lesník →
- Číslo popisné – 463
- Zařazení –
- Vztah k jiné osobě – kolega
- Jméno této osoby – Andrej
- Příjmení této osoby – → Koliba
- Rok nabytí gruntu – 1899
- Penežní dávky –
- Naturálni dávky –
- Roboty –
- Režijní podnikání, další majetky a práva –

Následující držitel 2:

- Grunt původně pustý/spálený –
- Jméno –
- Příjmení – → Novotný
- Povolání – čašník
- Číslo popisné – 490
- Zařazení –
- Vztah k jiné osobě –
- Jméno této osoby –
- Příjmení této osoby –
- Rok nabytí gruntu –
- Penežní dávky –
- Naturálni dávky –
- Roboty –
- Režijní podnikání, další majetky a práva –

Následující držitel 3:

- Grunt původně pustý/spálený –
- Jméno –
- Příjmení –
- Povolání –
- Číslo popisné –
- Zařazení –
- Vztah k jiné osobě –
- Jméno této osoby –
- Příjmení této osoby –
- Rok nabytí gruntu –
- Penežní dávky –
- Naturálni dávky –
- Roboty –
- Režijní podnikání, další majetky a práva –

Poznámka:

- Poznámka – Poznámka

## A.5 Nové záznamy z Lánové rejstříky

### A.5.1 Plne vyplnený formulár

Pozice:

- Pořadí scanu – 1
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Místo:

- Panství – Panstvo
- Obec – Nové Zámky
- Kategorie – kategória
- p;s;p/s – p
- Jméno gruntu – meno gruntu

Osedlý:

- Titul – bc
- Jméno – Roman
- Příjmení – Volko
- Alternativní příjmení – Volek
- Sirotek – áno
- Žid – áno
- Pohlaví – Muž

Sirotek po:

- Jméno – Marina
- Příjmení – Volková

Vdovec/Vdova po:

- Jméno – Alena
- Příjmení – Volková

Syn/Dcera koho

- Jméno – Marina
- Příjmení – Volková

Živnost:

- Druh živnosti – stolár
- Šenkovní mešťan – áno
- Jméno šenkovního domu – meno domu
- Mlynář – áno
- Mlýn – áno

Vinohrad:

- Jméno držitele – Radim
- Příjmení držitele – Škoda
- Rozloha – 123
- Název viniční hory/tratě – Názov trate

Pole:

- Počet kusů polí – 3
- 1. třída měric – 20
- 1. třída achtlů – 30
- 2. třída měric – 30
- 2. třída achtlů – 40
- 3. třída měric – 40
- 3. třída achtlů – 50

Jiná pole od:

- Jméno – Bonifác
- Příjmení – Janošek
- Počet kusů polí – 3
- 1. třída měric – 10
- 1. třída achtlů – 20
- 2. třída měric – 20
- 2. třída achtlů – 30
- 3. třída měric – 30
- 3. třída achtlů – 40

Jeho pole vlastní:

- Jméno – Klára
- Příjmení – Bukáčková
- Rok zpustnutí usedlosti – 1789
- Důvod zpustnutí usedlosti – požiar
- Rok zbudování usedlosti – 1775

Poznámka:

- Poznámka – Poznámka

### A.5.2 Čiastočne vyplnený formulár

Pozice:

- Pořadí scanu – 2
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Místo:

- Panství –
- Obec – Lhota
- Kategorie –
- p;s;p/s –
- Jméno gruntu – Menu gruntu 123

Osedlý:

- Titul –
- Jméno –
- Příjmení – Žilková
- Alternativní příjmení –
- Sirotek –
- Žid –
- Pohlaví – Žena

Sirotek po:

- Jméno –
- Příjmení –

Vdovec/Vdova po:

- Jméno –
- Příjmení –

Syn/Dcera koho

- Jméno – Jitka
- Příjmení – Hadáčová

Živnost:

- Druh živnosti – Pestovanie zeleniny
- Šenkovní mešťan –
- Jméno šenkovního domu –
- Mlynář –
- Mlýn –

Vinohrad:

- Jméno držitele –
- Příjmení držitele –
- Rozloha –
- Název viniční hory/tratě –

Pole:

- Počet kusů polí – 1
- 1. třída měric – 0
- 1. třída achtlů – 0
- 2. třída měric – 50
- 2. třída achtlů – 80
- 3. třída měric – 0
- 3. třída achtlů – 0

Jiná pole od:

- Jméno –
- Příjmení –
- Počet kusů polí –
- 1. třída měric –
- 1. třída achtlů –
- 2. třída měric –
- 2. třída achtlů –
- 3. třída měric –
- 3. třída achtlů –

Jeho pole vlastní:

- Jméno –
- Příjmení –
- Rok zpustnutí usedlosti –
- Důvod zpustnutí usedlosti –
- Rok zbudování usedlosti –

Poznámka:

- Poznámka – Poznámka

## A.6 Úprava záznamov z Lánové rejstříky

### A.6.1 Plne vyplnený formulár

Pozice:

- Pořadí scanu – 1
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Místo:

- Panství – Panstvo
- Obec – Nové Zámky
- Kategorie – kategória → iná kategória
- p;s;p/s – p → s
- Jméno gruntu – meno gruntu → názov 123

Osedlý:

- Titul – bc →
- Jméno – Roman
- Příjmení – Volko
- Alternativní příjmení – Volek →
- Sirotek – áno → nie
- Žid – áno → nie
- Pohlaví – Muž

Sirotek po:

- Jméno – Marina →
- Příjmení – Volková →

Vdovec/Vdova po:

- Jméno – Alena
- Příjmení – Volková

Syn/Dcera koho

- Jméno – Marina
- Příjmení – Volková

Živnost:

- Druh živnosti – stolár
- Šenkovní mešťan – áno → nie
- Jméno šenkovního domu – meno domu →
- Mlynář – áno → nie
- Mlýn – áno

Vinohrad:

- Jméno držitele – Radim → Vladimír
- Příjmení držitele – Škoda
- Rozloha – 123 → 119
- Název viniční hory/tratě – Názov trate → Iný názov

Pole:

- Počet kusů polí – 3 → 5
- 1. třída měric – 20 → 15
- 1. třída achtlů – 30
- 2. třída měric – 30
- 2. třída achtlů – 40
- 3. třída měric – 40
- 3. třída achtlů – 50

Jiná pole od:

- Jméno – Bonifác
- Příjmení – Janošek → Janoš
- Počet kusů polí – 3
- 1. třída měric – 10
- 1. třída achtlů – 20
- 2. třída měric – 20
- 2. třída achtlů – 30
- 3. třída měric – 30
- 3. třída achtlů – 40

Jeho pole vlastní:

- Jméno – Klára
- Příjmení – Bukáčková → Bukáčiková
- Rok zpustnutí usedlosti – 1789 → 1788
- Důvod zpustnutí usedlosti – požiar
- Rok zbudování usedlosti – 1775 → 1774

Poznámka:

- Poznámka – Poznámka → Poznámka 2

### A.6.2 Čiastočne vyplnený formulár

Pozice:

- Pořadí scanu – 2
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Místo:

- Panství –
- Obec – Lhota → Lehota
- Kategorie – → kategória 5
- p;s;p/s –
- Jméno gruntu – Menu gruntu 123 → Meno gruntu

Osedlý:

- Titul –
- Jméno – → Agáta
- Příjmení – Žilková
- Alternativní příjmení –
- Sirotek – → áno
- Žid –
- Pohlaví – Žena

Sirotek po:

- Jméno – → Bohuslav
- Příjmení – → Žilka

Vdovec/Vdova po:

- Jméno –
- Příjmení –

Syn/Dcera koho

- Jméno – Jitka → Bohuslav
- Příjmení – Hadáčová → Žilka

Živnost:

- Druh živnosti – Pestovanie zeleniny
- Šenkovní mešťan –
- Jméno šenkovního domu –
- Mlynář –
- Mlýn – → áno

Vinohrad:

- Jméno držitele –
- Příjmení držitele –
- Rozloha –
- Název viniční hory/tratě –

Pole:

- Počet kusů polí – 1 → 2
- 1. třída měric – 0
- 1. třída achtlů – 0
- 2. třída měric – 50
- 2. třída achtlů – 80
- 3. třída měric – 32
- 3. třída achtlů – 45

Jiná pole od:

- Jméno –
- Příjmení –
- Počet kusů polí –
- 1. třída měric –
- 1. třída achtlů –
- 2. třída měric –
- 2. třída achtlů –
- 3. třída měric –
- 3. třída achtlů –

Jeho pole vlastní:

- Jméno –
- Příjmení –
- Rok zpustnutí usedlosti –
- Důvod zpustnutí usedlosti –
- Rok zbudování usedlosti –

Poznámka:

- Poznámka – Poznámka

## A.7 Nové záznamy z Poddanské přiznávací fase

### A.7.1 Plne vyplnený formulár

Pozice:

- Pořadí scanu – 1
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Umístnení:

- Držitelem je obec nebo vrchnost – áno
- Panstvo – Panstvo 123
- Obec – Plchov

Držitel v době 2. lánové privatizace:

- Jméno – Kryštof
- Příjmení – Urbán
- Pohlaví – Muž
- Povolání – Drotár

Příbuzný držitele:

- Vztah k držiteli – syn
- Jméno – Vilém
- Příjmení – Urbán

Grunt:

- Velikost (měřice) – 20
- Velikost (achtel) – 30
- Zaseto a vinice (měřice) – 10
- Zaseto a vinice (achtel) – 15
- Dražebnostní číslo – 531

Současný držitel:

- Jméno – Samuel
- Příjmení – Marvan
- Pohlaví – Muž
- Povolání – Spisovateľ

Příbuzný současného držitele:

- Vztah k současnému držiteli – sestra
- Jméno – Miroslava
- Příjmení – Marvanová

Pole:

- Počet – 5
- Výměra (měřice) – 40
- Výměra (achtel) – 50

Pastviny:

- Výměra (měřice) – 5
- Výměra (achtel) – 10

Záhrady:

- Výměra (měřice) – 27
- Výměra (achtel) – 45

Louky:

- Název – Názov lúky
- Počet – 1
- Fůri sena – 2
- Fůri otava – 3

Vinice:

- Výměra (měřice) – 0
- Výměra (achtel) – 0

Lesy:

- Výměra (měřice) – 11
- Výměra (achtel) – 13

Rybníky:

- Kapři – 0
- Násadové – 0
- Pstruzi – 15

Ostatní:

- Mlýn – áno
- Pila – áno
- Olejárna – áno
- Valcha – áno
- Papírna – áno
- Řemesla, způsob obživy – chov

Poznámka:

- Poznámka – Poznámka

### A.7.2 Čiastočne vyplnený formulár

Pozice:

- Pořadí scanu – 15
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 2
- Jazyk – Staročeský

Umístnení:

- Držitelem je obec nebo vrchnost –
- Panstvo –
- Obec –

Držitel v době 2. lánové privatizace:

- Jméno – Blanka
- Příjmení – Vostrá
- Pohlaví – Žena
- Povolání – Sochárka

Příbuzný držitele:

- Vztah k držiteli – dcéra
- Jméno –
- Příjmení – Vostrá

Grunt:

- Velikost (měřice) – 28
- Velikost (achtel) – 35
- Zaseto a vinice (měřice) –
- Zaseto a vinice (achtel) –
- Dražebnostní číslo –

Současný držitel:

- Jméno – Ludvík
- Příjmení – Turoň
- Pohlaví – Muž
- Povolání –

Příbuzný současného držitele:

- Vztah k současnému držiteli –
- Jméno –
- Příjmení –

Pole:

- Počet – 0
- Výměra (měřice) – 0
- Výměra (achtel) – 0

Pastviny:

- Výměra (měřice) –
- Výměra (achtel) –

Záhrady:

- Výměra (měřice) –
- Výměra (achtel) –

Louky:

- Název –
- Počet – 3
- Fůri sena – 11
- Fůri otava –

Vinice:

- Výměra (měřice) – 56

- Výměra (achtel) – 87

Lesy:

- Výměra (měřice) – 11
- Výměra (achtel) – 45

Rybníky:

- Kapři –
- Násadové –
- Pstruzi –

Ostatní:

- Mlýn – nie
- Pila – nie
- Olejárna – nie
- Valcha – nie
- Papírna – nie
- Řemesla, způsob obživy –

Poznámka:

- Poznámka – Poznámka

## A.8 Úprava záznamov z Poddanské přiznávací fase

### A.8.1 Plne vyplnený formulár

Pozice:

- Pořadí scanu – 1
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 1
- Jazyk – Staročeský

Umístnení:

- Držitelem je obec nebo vrchnost – áno → nie
- Panstvo – Panstvo 123 →
- Obec – Plchov →

Držitel v době 2. lánové privatizace:

- Jméno – Kryštof
- Příjmení – Urbán → Urban
- Pohlaví – Muž
- Povolání – Drotár

Příbuzný držitele:

- Vztah k držiteli – syn
- Jméno – Vilém → Vilhem
- Příjmení – Urbán → Urban

Grunt:

- Velikost (měřice) – 20 → 23
- Velikost (achtel) – 30 → 33
- Zaseto a vinice (měřice) – 10 → 13
- Zaseto a vinice (achtel) – 15 → 18
- Dražebnostní číslo – 531

Současný držitel:

- Jméno – Samuel
- Příjmení – Marvan
- Pohlaví – Muž
- Povolání – Spisovateľ

Příbuzný současného držitele:

- Vztah k současnému držiteli – sestra → sesternica
- Jméno – Miroslava
- Příjmení – Marvanová

Pole:

- Počet – 5 → 4
- Výměra (měřice) – 40
- Výměra (achtel) – 50

Pastviny:

- Výměra (měřice) – 5
- Výměra (achtel) – 10

Záhrady:

- Výměra (měřice) – 27 → 22
- Výměra (achtel) – 45 → 40

Louky:

- Název – Názov lúky → Iný názov lúky
- Počet – 1 → 3
- Fůri sena – 2 → 5
- Fůri otava – 3 → 5

Vinice:

- Výměra (měřice) – 0 → 4
- Výměra (achtel) – 0 → 6

Lesy:

- Výměra (měřice) – 11
- Výměra (achtel) – 13

Rybníky:

- Kapři – 0 → 11
- Násadové – 0 → 7
- Pstruzi – 15 → 14

Ostatní:

- Mlýn – áno → nie
- Pila – áno → nie
- Olejárna – áno → nie
- Valcha – áno → nie
- Papírna – áno → nie
- Řemesla, způsob obživy – chov →

Poznámka:

- Poznámka – Poznámka

### A.8.2 Čiastočne vyplnený formulár

Pozice:

- Pořadí scanu – 15
- Rozložení na scanu – Přes celé
- Pořadí záznamu – 2
- Jazyk – Staročeský

Umístnení:

- Držitelem je obec nebo vrchnost – → áno
- Panstvo – Panstvo
- Obec –

Držitel v době 2. lánové privatizace:

- Jméno – Blanka → Petr
- Příjmení – Vostrá → Vostrí
- Pohlaví – Žena → Muž
- Povolání – Sochárka → Sochár

Příbuzný držitele:

- Vztah k držiteli – dcéra
- Jméno – → Vlasta
- Příjmení – Vostrá

Grunt:

- Velikost (měřice) – 28
- Velikost (achtel) – 35
- Zaseto a vinice (měřice) –
- Zaseto a vinice (achtel) –
- Dražebnostní číslo – → 331

Současný držitel:

- Jméno – Ludvík
- Příjmení – Turoň
- Pohlaví – Muž
- Povolání – → Obchodník

Příbuzný současného držitele:

- Vztah k současnému držiteli – → otec
- Jméno – → Roman
- Příjmení – → Turoň

Pole:

- Počet – 0 → 1
- Výměra (měřice) – 0 → 26
- Výměra (achtel) – 0 → 34

Pastviny:

- Výměra (měřice) –
- Výměra (achtel) –

Záhrady:

- Výměra (měřice) – → 11
- Výměra (achtel) – → 22

Louky:

- Název – → Názov
- Počet – 3
- Fůri sena – 11
- Fůri otava – → 0

Vinice:

- Výměra (měřice) – 56
- Výměra (achtel) – 87

Lesy:

- Výměra (měřice) – 11
- Výměra (achtel) – 45

Rybníky:

- Kapři – → 0
- Násadové – → 0
- Pstruzi – → 0

Ostatní:

- Mlýn – nie → áno
- Pila – nie
- Olejárna – nie
- Valcha – nie
- Papírna – nie
- Řemesla, způsob obživy –

Poznámka:

- Poznámka – Poznámka