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

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

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


# DÁVKOVÉ ZPRACOVÁNÍ GENEALOGICKÝCH DAT

**BATCH PROCESSING OF GENEALOGICAL DATA**

**BAKALÁŘSKÁ PRÁCE**
**BACHELOR'S THESIS**


**AUTOR PRÁCE** — **ADAM JANDA**
**AUTHOR**

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



**BRNO 2023**

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

# Zadání bakalářské práce

144766

Ústav: Ústav inteligentních systémů (UITS)
Student: **Janda Adam**
Program: Informační technologie
Specializace: Informační technologie

Název: **Dávkové zpracování genealogických dat**
Kategorie: Databáze
Akademický rok: 2022/23

Zadání:

1. Seznamte se s problematikou práce s genealogickými daty a s existující databází přepisovaných matričních záznamů ČR. Tato výchozí databáze vzniká v rámci projektu na FIT.
2. Navrhněte systém šablon pro import a export dat do a z genealogické databáze. Šablony slouží jako alternativa k uživatelskému rozhraní pro zadávání dat a získání výsledků vyhledávání. Prostřednictvím šablon je možné vkládat nové záznamy, upravovat stávající či exportovat vybraná data.
3. Navrhněte a implementujte systém pro transformaci záznamů z připravených šablon do databáze a zpětně z databáze do šablon. Systém bude fungovat jako samostatný modul a současně bude zakomponován do webového rozhraní k databázi matričních záznamů.
4. Na reálných nebo demonstračních datech otestujte systém a diskutujte jeho další vývoj.

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.


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: 1.11.2022
Termín pro odevzdání: 10.5.2023
Datum schválení: 3.11.2022

# Abstrakt

Táto práca popisuje návrh, vývoj a implementáciu dávkového spracovania matričných záznamov pomocou systému šablón. Proces dávkového spracovania záznamov je rozšírením projektu DEMoS, ktorý sa zaoberá digitalizáciou historických matričných zápisov. Pripojenie navrhnutého modulu k webovému rozhraniu systému DEMoS, prináša možnosť hromadne vkladať nové záznamy, prípadne ich dávkovú editáciu. Súčasťou riešenia je vkladanie dát, extrahovanie matričných dát do súboru CSV uložených v databáze a exportovanie vzorového súboru CSV. Pre každý typ matričného záznamu (záznam narodenia, úmrtia a sobášu) bolo navrhnutých dokopy 15 šablón. Úpravou alebo rozšírením týchto šablón, je možné proces spracovania prispôsobiť architektonickej zmene štruktúry databáze, ktorá sa deje z dôvodu neucelených formátov matričných zápisov.



# Abstract

This thesis describes the design, development, and implementation of batch processing of parish records using a template system. The batch processing of records is an extension of the DEMoS project, which deals with the digitization of historical parish records. The connection of the designed module to the web interface brings the possibility of bulk addition of new records or batch editing. Part of the solution is data entry, the extraction of parish data into a CSV file stored in the database. A total of 15 templates were designed for each type of parish record (birth, death, and marriage records). By modifying or expanding these templates, batch processing can be adapted to the architectural change in the database structure due to the incompleteness of parish register formats.



# Kľúčové slová

matričné záznamy, šablóny, dávkové spracovanie, import, export

# Keywords

parish records, templates, batch processing, import, export


# Citácia

JANDA, Adam. *Dávkové zpracování genealogických dat.* Brno, 2023. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Radek Kočí, Ph.D.

# Dávkové zpracování genealogických dat

## Prehlásenie

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

.....
Adam Janda
8. mája 2023


## Poďakovanie

Rád by som poďakoval predovšetkým vedúcemu práce Ing. Radkovi Kočímu, Ph.D. za odbornú pomoc, rady pri písaní práce a za vedenie práce. Ďalej by som chcel poďakovať rodine a priateľom za morálnu podporu, a pomoc počas tvorby práce, a počas štúdia na Vysokém učení technickém v Brně, Fakulty informačních technologií.

# Obsah

# Kapitola 1

# Úvod

Genealogické záznamy nám v dnešnom svete umožňujú zjednodušený pohľad do minulosti a života našich predkov. Vďaka týmto záznamom vieme spojiť jednotlivých ľudí, ich základné životné míľniky a určiť hrubý odhad ich životného príbehu. Záznamy taktiež sprístupňujú schopnosť vyčítať rôzne vzťahy medzi ľuďmi, nielen na základe rodinného príbuzenstva, ale aj ich postavenie v spoločnosti prostredníctvom ich pracovného titulu. Medzi spomenuté míľniky, z pohľadu matrík, patrí narodenie, uzavretie sňatku ale aj úmrtie, roztrúsené v rôznych historických listinách a písomnostiach.

Keďže sa zakladanie matričných záznamov vedie od stredoveku, v dnešnej dobe je ich počet veľmi veľký a ich vedenie bolo prevažne vo farských matričných knihách uložených po rôznych kútoch krajiny. Vyhľadávanie v týchto veľmi starých, a zväčma hrubých knihách prináša niekoľko obtiažností, ako napríklad množstvo času potrebného k nájdeniu konkrétnej informácie, vyťaženosť a dostupnosť matričných úradov, alebo samotná krehkosť historických kníh. Kvôli týmto a mnoho ďalším problémom a požiadavkám, sa v poslednej dobe inštitúcie snažia tieto matričné knihy digitalizovať, čím odomykajú pomyselné dvere k vedomostiam dávnych dôb.

Digitalizácia historických matričných záznamov prináša možnosť hlbšej a rýchlej analýzy historických súvislostí, ktoré nás zaujímajú. Okrem analýzy získavame digitalizáciou schopnosť informácie normalizovať, v prípade potreby korektúrou opraviť a v najdôležitejšej úlohe ukladať na jednom centrálnom mieste, vďaka čomu nedochádza k zbytočnej duplikácii a strate historických dát, tak ako sa stávalo v minulosti. Takto digitálne spracované dáta vieme jednoducho previazať s iným typom historických záznamov, ako napríklad pozemkové urbáre, z ktorých pomocou dokážeme historickú analýzu obohatiť a priniesť väčší význam jej výskumu.

Na Fakulte informačních technologií VUT vznikol, v spolupráci s Filozofickou fakultou Masarykovej univerzity v Brne, systém DEMoS (z angl. *Database of Early Modern Sources*) pre uchovávanie a následné spracovanie prepisov matričných záznamov. Systém je stále v aktívnom vývoji a jeho architektúra sa prispôsobuje novým formám zápisu historických, matričných udalostí. Zmeny architektúry sa prejavujú tým, že sa do sústavy entít vkladajú nové entity, alebo do stávajúcich sa pridávajú nové atribúty, prípadne sa odstraňujú, a premiestňujú. Pretože genealogických dát je veľké množstvo, a vkladanie ručne po jednej zaberá veľmi veľa času, funkcionalita má veľké obmedzenia, a ešte väčší priestor na zlepšenie.

Riešením tohto problému je dávkové spracovanie matričných informácií. Spojením stále sa meniacej architektúry a veľkého množstva dát, systém DEMoS potrebuje efektívny popis vzťahov systémových entít, ktorý sa bude dať použiť aj ako popis pre mechanizmus dávkového vkladania, extrahovania a aktualizácie dát. Na proces vkladania nových záznamov,

prípadnú editáciu už vytvorených, bude nadväzovať navrhované riešenie popísané v tejto práci a umožní hromadné akcie pre niekoľko záznamov naraz, tak aby vedelo rýchlo reagovať na zmeny architektúry systému.

Obsahom práce bude zoznámenie sa s problematikou a odbornou látkou. Na získané základy naviaže kapitola návrhu šablón, v ktorej sa zadefinuje syntax a prepojenie šablónových entít. Následne na to sa priblíži mechanizmus praktickej časti, ktorou bude dávkový import a export dát, použitím navrhnutých šablón. Poslednou časťou je riešenie zakomponovať do systému DEMoS a overenie validnosti procesu spracovania dát. Práca si teda dáva za cieľ navrhnutie a implementáciu systému šablón definujúcich logické vzťahy medzi entitami.

# Kapitola 2

# Štúdium problematiky

Pred samotným návrhom šablón, a implementáciou navrhnutého riešenia, práca obsahuje teoretický výklad potrebný ku správnemu porozumeniu textu, s ktorým pracuje existujúci systém DEMoS, a teoretickou časťou problému tohto textu.

V nasledujúcich podkapitolách a sekciách sa definujú odborné pojmy, hneď nato sa nastieni systém DEMoS a jeho štruktúra. Zakončením tejto časti je vysvetlenie s akými dátami pracuje existujúci systém a súčasné možnosti spracovania daných údajov.

## 2.1 Terminológia

Táto podkapitola je zameraná na teoretické poznatky, s ktorými sa bolo potrebné stotožniť pre pochopovenie odborných pojmou. Sekcie kapitoly vysvetľujú pojem genealógia, odkiaľ slovo pochádza a čo popisuje. V ďalšej sekcii sa vysvetlí pojem matrika a jej historický vývoj. Poslednou sekciou bude časť, v ktorej sa vysvetlí proces dávkového spracovania dát.

### 2.1.1 Genealógia

Slovo genealógia sa skladá z dvoch cudzích gréckych slov *genos* (rod, pokolenie) a *logos* (náuka, veda). Genealógia je pomocná vedná disciplína, zaoberajúca sa štúdiom histórie rodiny, najmä cez historické dokumenty, k objasneniu vzťahov medzi konkrétnymi ľuďmi a ich rodinami. Okrem vzťahov skúma aj následky, ktoré nasledovali po ich vzniknutí po rôznych stránkach, či už historických, sociálnych, biologických, alebo iných [7, 18].

### 2.1.2 Matrika

V súčasnosti pojem matrika je definovaná ako verejná kniha obsahujúca osobné údaje občanov. Do matriky sa vkladajú záznamy o narodeniach, úmrtiach, manželstvách a vzniku registrovaného partnerstvia občanov Českej republiky, ako ustanovuje právny predpis *zákona č. 301/2000 Sb. § 1o matrikách, jménu a příjmení a o změně některých souvisejících zákonů* [15].

Vedenie matričných udalostí a matričných skutočností má dlhú historickú tradíciu. Na území Českej republiky to mali na starosti farnosti z hruba od 14. storočia. Zo začiatku sa jednalo o zoznamy pokrstených, birmovaných a zosobášených. Neskôr sa k týmto zoznamom pridal aj zoznam zomrelých. Vytváranie týchto zoznamov bolo nepovinné a povinnosť katolíckym kňazom uložil napokon Tridentský koncil v roku 1563, ktorá sa ale začala prísnejšie dodržiavať až na prelome 16. a 17. storočia. Samotný zápis dlhú dobu nebol nijak

centrálne určený, preto sa jeho formy a obsah údajov líšia z farnosti k farnosti. Jednotné pravidlá zápisu matrík pôrodu, sobášu a úmrtia priniesol až patent císara Jozefa II. z dňa 20. februára 1784. Okrem zjednotených pravidiel, patent, rozšíril povinnosť protestantským farnostiam a židovkým rabínom viesť tzv. registre pôrodov, sobášov a úmrtí. Aj po prijatí patentu, vedenie matrík zostávalo pod záštitou cirkevných orgánov až do 1.1.1950, kedy ich vedenie prebral štát na základe zákona č. 268/1949 Sb., o matrikách [1, 2, 6].

### 2.1.3 Dávkové spracovanie

Dávkové spracovanie je mechanizmus, kedy sa spracúva veľké množstvo dát z jedného zdroja v jeden okamih. Po spustení dávkového spracovania, proces prebieha na pozadí bez prítomnosti interakcie od užívateľa [14].

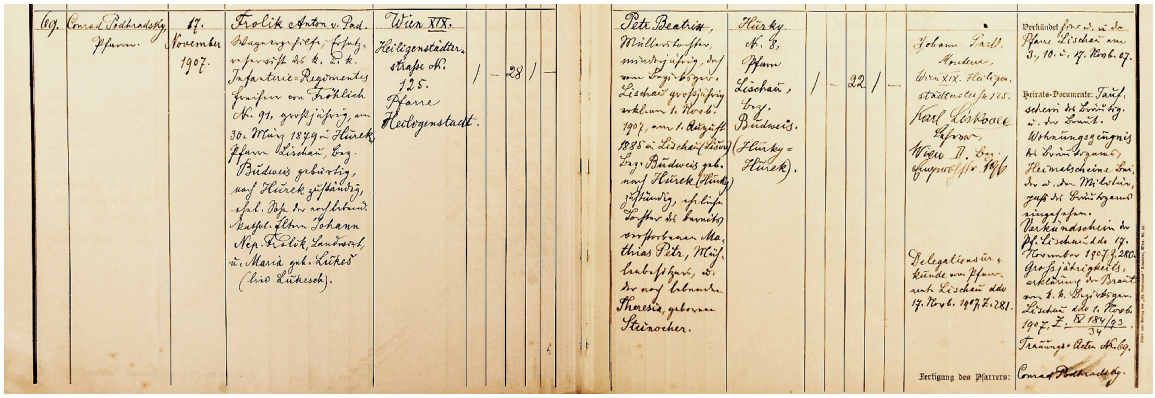
Úloha procesu sa nazýva tzv. dávková úloha (angl. *batch job*). Dávková úloha nemá žiadne užívateľské prostredie, a všetky vstupy sú predkladané procesu dávkového spracovania prostredníctvom príkazových parametrov, skriptov, konfiguračných súborov, alebo dát [11].

## 2.2 Archívne identifikovanie záznamov

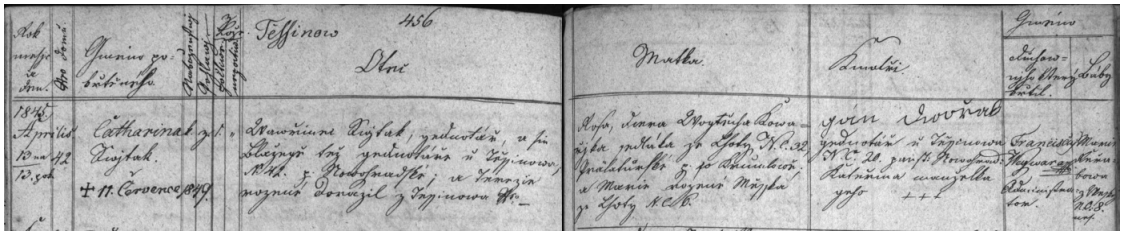
Matričné udalosti sú zapisované do príslušnej matriky na lokálnom úrade. Každá lokálna matrika sa identifikuje podľa názvu alebo skratky, ďalej signatúry a fondu, určeného podľa archívneho úradu.

Systém DEMoS pracuje s digitálnou verziou matričných záznamov, vo forme digitálnej fotografie tzv. skenu. Sken môže zachytávať jednu alebo dve strany matriky. Počet zachytených strán skenu je určený podľa typu záznamu, keďže niektoré záznamy sú vedené cez dve strany horizontálne, tak ako je zobrazené na obrázkoch 2.1 a 2.2.

Matričné záznamy sú na skene identifikované fyzickou polohou (vľavo, vpravo, v strede), poradím skenu (číslo poradia vyhotovenia pre danú matriku) a poradím matričného záznamu (poradové číslo údaju v matrike). Komplentná identifikácia matričnej udalosti v celom systéme DEMoS sa skladá z názvu/skratky matriky, fondu matričného úradu, signatúry, poriadia skenu, poradia záznamu a rozloženia na skene [12].



Obr. 2.1: Matričný zápis sobášu z roku 1907 [13].



Obr. 2.2: Rímskokatolický farný úrad Jílovice, matrika narodení 1804-1865 [19].

## 2.3 Význam systému DEMoS

Systém DEMoS sa stará o uloženie digitalizovaných dát a udržiavanie väzieb medzi jednotlivými osobami spomenutých v matričných záznamoch. Jedná sa teda o relačnú databázu, ktorej hlavné entity matričnej udalosti môžu mať okolo 30 atribútov, ktoré ich popisujú. Kompletný záznam jedného typu dosahuje počtu približne 230 atribútov, rozdelených po menších častiach zodpovedajúcim aktérom udalosti, ktoré nie vždy obsahujú nejakú hodnotu.

Relatívne vysoký rozdiel medzi počtom atribútov matričnej udalosti, a kompletného záznamu je ten, že každý typ záznamu má rozdielny počet informácii, a aktérov v daných udalostiach. Medzi ďalší rozdiel v počte atribútov zohráva hlavne rôzna forma zápisov v jednotlivých typoch matrík. Tak ako bolo spomenuté v sekcii 2.1.2, formáty zápisov si dlhú dobu určovali jednotlivé farnosti, preto spojením každého rozdielu vzniká tak vysoký počet atribútov.

Hlavným cieľom systému nie je len uchovávanie matričných záznamov, ale v budúcnosti, aj pripojenie zdrojov ako sčítanie ľudu, alebo katastrálnych záznamov [12].

## 2.4 Architektúra systému DEMoS

Model systému DEMoS spočíva v relačnej databáze a webovej aplikácie. Tento model sa v softvérovom inžinierstve nazýva klient–server a bližšie priblíženie ku konkrétnym technológiám serveru je zmienené v kapitole 5.1.

Model klient–server je založený na komunikácii medzi dvoma nezávislými aplikačnými procesmi, klient a server. Aplikačným procesom klienta rozumieme hocijaký proces alebo aplikáciu, ktorá zasiela požiadavky na získanie služby cieľového aplikačného procesu serveru. Server je teda proces, ktorý poskytuje služby pre klienta. Oba tieto procesy môžu byť na rovnakom počítači alebo fyzicky na rôznych strojoch. Ak klientských procesov je viacej na rôznych počítačoch, server môže obsluhovať ich požiadavky na služby a komunikácia prebieha prostredníctvom počítačovej siete. Klientský proces môže pokladať požiadavky na viacerých serveroch [25].

### 2.4.1 Relačná databáza

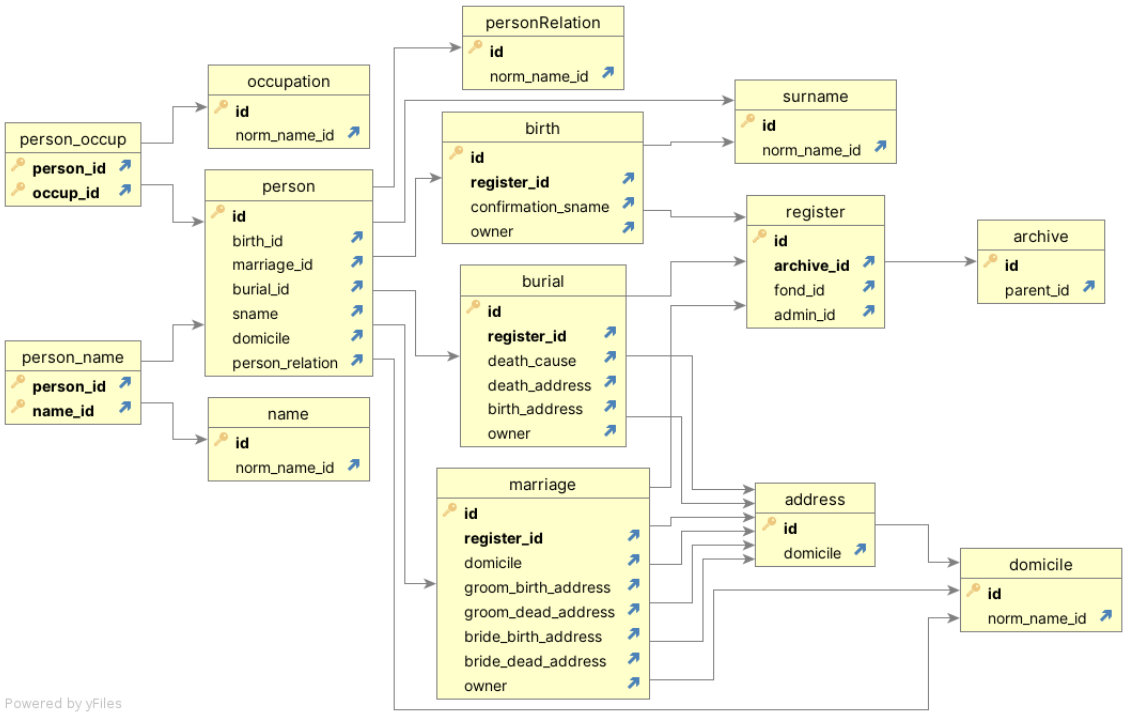
Základom relačnej databáze je model založený na relačných dátach. Dáta organizuje do riadkov a stĺpcov, ktoré spolu tvoria tabuľku. Tabuľky obsahujú unikátne identifikátory, tzv. primárne kľúče (angl. *primary keys*) a cudzie kľúče (angl. *foreign keys*). Vďaka týmto identifikátorom, je možné tabuľky spájať a vytvárať medzi nimi rôzne vzťahy.

Nad dátami je možné pokladať otázky pomocou štruktúrovaného dopytovacieho jazyka SQL (z angl. *Structured Query Language*). Tak ako relačná databáza, tak aj jazyk SQL je založený na relačnej algebre, a n-ticovom relačnom počte. SQL pozostáva z mnohých typov príkazov. Príkazy jazyka SQL slúžia k vytváraniu, a modifikovaniu dát, a rovnako tak, aj k definovaniu štruktúry databáze [21].

### Databázové tabuľky systému DEMoS

Z dôvodu normalizácie dát, boli tabuľky databáze rozdelené na tabuľky udržujúce špecifické informácie o matričnom zázname a na tabuľky, ktoré obsahujú všeobecne známe dáta. Základná štruktúra architektúry databáze DEMoS je vykreslená na obrázku 2.3. Medzi tabuľky, ukladajúce špecifické informácie, patria *birth*, *marriage*, *burial* a *person*, ktoré sú rozšírené o väzobné tabuľky medzi entitami. Spolu tvoria základnú kostru pre ukladania matričných záznamov.

Počet tabuliek, v ktorých sú uložené všeobecné informácie, je sedem. Medzi ne patria tabuľky mien, priezvisk, názvov vzťahov osôb, názvov povolaní, názvov príčiny úmrtia, a názvov bydlísk, a adries. Tieto tabuľky sú používané ako tzv. slovníky. Okrem týchto dvoch druhov tabuliek, databáza disponuje ďalšími, ktoré slúžia na uchovanie informácii o archívnych úradoch a celkoch, rovnako tak aj o užívateľoch systému DEMoS [12].




Obr. 2.3: Základná štruktúra databáze DEMoS.

## 2.4.2 Webová aplikácia

Webovou aplikáciou rozumieme počítačový program, ktorý je využívaný webovými prehliadačmi a inými webovými aplikáciami. Ponúkajú rozdielne služby a funkcionality na celom internete. Aplikačné dáta sú uložené na webovom serveri a ich získanie inicializuje klientský webový prehliadač, tak aby spĺňoval definíciu modelu klient–server.

Vykreslenie webovej aplikácie na webovej stránke definujú programovacie jazyky, ktoré sú podporované webovými prehliadačmi, ako napríklad HTML (angl. *HyperText Markup Language*) a CSS (angl. *Cascading Style Sheets*). Obsah webovej stránky môže byť statického alebo dynamického druhu. Pre statický obsah webovej stránky nie je potrebný žiadny ďalší výpočet na strane webového serveru. Dynamický obsah webových stránok vyžaduje výpočet alebo vykonanie logického procesu na strane serveru. Tieto akcie sú definované programovacími jazykmi, ako napríklad Java alebo PHP (angl. *PHP: Hypertext Preprocessor*), alebo inými systémovými nástrojmi [20].

Webové rozhranie systému DEMoS má na starosti aplikačný rámec Nette, písaný v programovacom jazyku PHP. Logiku a štruktúru webovej aplikácie systému DEMoS je detailnejšie rozpísaná v neskoršej kapitole 5.2.

## 2.5 Typy matričných záznamov systému DEMoS

Systém DEMoS rozoznáva tri typy matričných záznamov. Sú nimi záznamy o narodení, o sobáši a o úmrtí. Každý typ záznamu má špeciálne atribúty, ktoré sa v ňom vyskytujú a na záznam sa viažu rôzne druhy spríbuznených vzťahov.

Unikátnym identifikátorom pre daný typ matričného záznamu v konkrétnej matričnej knihe je kombinácia poradia scanu, poradie záznamu a rozloženie na skenu. Táto kombinácia presne identifikuje polohu v matričnej knihe a platí pre všetky typy záznamov. Detailnejší popis jednotlivých atribútov a vzťahov medzi týmito atribútmi je bližšie priblížený v samostatných podkapitolách.

Často používaným typom matričného záznamu je jeho fyzická podoba v knihách. Z tohto dôvodu často dochádza k prekrývaniu informácií. Príkladom môžu byť krstné mená, priezviska alebo názvy pracovných titulov a zamestnaní. Používaný systém s možnosťou prekrytia informácií počíta a dané vstupy normalizuje.

### 2.5.1 Záznam narodenia

Hlavná časť záznamu narodenia obsahuje atribúty dátumu, času a miesta krstu a narodenia, pohlavie dieťaťa, jazyk záznamu, v ktorom bol zapísaný a iné. Na tento hlavný záznam sa viaže birmovné meno, osoby vo vzťahu k narodeniu a samotná matričná kniha.

Medzi základné osoby, ktoré majú vzťah k záznamu narodenia, je v prvom rade samotné novonarodené dieťa, matka a otec dieťaťa. K ďalším osobám patria krstiteľ, pôrodná bába, kmotrovia dieťaťa a príbuzní rodičov.

### 2.5.2 Záznam sobášu

Záznam sobášu drží informácie o ohláškach (až do tretieho stupňa), dátum sobášu a stupeň príbuznosti novomanželov. Ďalej medzi atribúty hlavnej časti sú vek novomanželov a to konkrétne rozdelené na roky, mesiace a dni. Hlavný záznam ešte rozširujú atribúty popisujúce adresy narodenia a úmrtia jednotlivo pre ženícha a nevestu, a pár ďalších atribútov.

Oproti záznamu narodenia, sobáš nemá unikátne definovú hlavnú osobu zaväzujúcu sa k hlavnému záznamu, lebo v záznamoch sa nachádzajú klasickí novomanželia, ale aj vdovci a vdovy. Osoby, ktoré figurujú v tomto type matričnej informácie sú, už vyššie spomenutý, ženích, nevesta, po prípade vdova a vdovec, oddávajúca osoba (bežne sa jednalo o kňaza), rodičia novomanželov a iní príbuzní.

### 2.5.3 Záznam úmrtia

Pri úmrtí sa evidujú dátumy úmrtia a samotného smútočného obradu, miesto pochovania, informácia či zomrelý bol mŕtvonarodený, adresa narodenia a úmrtia, a ďalšie vedľajšie informácie.

Ako v prípade záznamu narodenia, u záznamu úmrtia sa dokáže definovať konkrétna osoba, ktorá sa označuje za hlavnú osobu záznamu. Matričné záznamy obsahujú aj osobu, ktorá obzrela zomrelého pred samotným pohrebom, zaopatrovateľa zomrelého, osobu, ktorá zomrelého pochovala a samozrejme aj spríbuznené osoby zomrelého, ako je otec, matka, manžel, manželka, syn a dcéra.

## 2.6 Spracovanie záznamov

V súčasnej dobe je vkladanie nových záznamov a ich úprava, tak ako už bolo spomenuté v úvode, vykonávaná ručne cez webové rozhranie systému. Ďalšou možnosťou vloženia nových záznamov je použitie migračných skriptov, ktoré dokážu len vkladať záznamy. Chýba im schopnosť existujúce záznamy automaticky modifikovať.

# Kapitola 3

# Návrh šablón

Dôležitou časťou tejto bakalárskej práce je návrh systému šablón. Nasledujúca kapitola postupne vysvetlí dôvod výberu riešenia pomocou šablón, ďalej zadanie požiadaviek na šablóny a analýzu formátu fyzického uloženia. Kapitola pokračuje zosumerizovaním ideí, ktoré sa týkajú návrhu, a nakoniec približuje štruktúru, a syntax šablón.

## 3.1 Príčina výberu šablón

Šablóny slúžia k definícii väzby medzi matričnými záznamami a tabuľkami databáze. Vďaka tejto väzbe vie aplikácia mapovať vstupné dáta, zo súboru CSV do príslušných tabuliek. Ďalší význam šablón spočíva v tom, že ich štruktúra bude jednoducho a rýchlo upraviteľná pre prípadné zmeny databázovej architektúry systému DEMoS. Vytvorený systém šablón bude mať okrem zjednodušeného prispôsobenie sa zmenám, aj ďalšie dvojité využitie. Prvým bude schopnosť systému vo väčších množstvách ovládať vkladanie nových a úpravu existujúcich dát v jeden okamih. Toto využitie bude mať za následok zefektívnenie práce so systémom DEMoS a prácu so spracovaním genealogických dát. Druhým využitím bude pridanie užívateľského rozhrania pre ovládanie procesu dávkového spracovania systémovému administrátorovi bez nutnosti zasahovať do logiky samotného procesu. Výsledkom bude jednotné ovládanie pre každý typ matričnej udalosti.

## 3.2 Požiadavky na šablóny

Pred návrhom konkrétneho riešenia syntaxe šablón, je vhodné zadefinovať povinnú funkcionalitu, ktorú budú zastrešovať a rovnako tak aj očakávania, ktoré by mali napĺňať.

V prvom rade musia vedieť šablóny poskytovať jednoduchý, prehľadný a rýchly spôsob, ako reagovať na architektonické zmeny, stále sa vyvíjajúcej, databáze systému DEMoS. Ďalej musia vedieť popisovať vo vyššej abstrakcii vzťahy logických entít systému. Spomenuté požiadavky budú simulovať užívateľské rozhranie pre systémového architekta. Jedná sa teda o vývojársky nástroj.

V druhom rade je potrebné, aby vedeli prepojiť väzby medzi tabuľkou, ktorú vyplnil užívateľ, a systémom. Znamená to, že bude potrebné aby popisovali mapovanie celku dát do normalizovaného formátu databázy. Toto kritérium umožní aplikácii dávkovo spracovať dáta, ktoré sú určené k vloženiu.

V poslednej rade je žiaduce, aby šablóny vedeli zrekonštruovať ľubovoľnú množinu vložených dát užívateľom, z normalizovaného stavu do stavu pred vložením.

## 3.3 Analýza prostriedkov

Uskutočnená analýza sa venovala prostriedkom alebo nosičom šablón, v ktorých budú šablóny uložené. Do porovnania sa budú dávať praxou štandardizovaný textový súbor vo formáte JSON (z angl. *JavaScript Object Notation*) a textový súbor vo formáte prostý text (z angl. *plain text*)[8].

Z požiadaviek vyplýva, že by šablóny mali byť jednoducho a rýchlo upraviteľné. Oba z porovnávaných formátov sú typu text a teda je ich možné upravovať pomocou bežného systémového textového editora. Táto vlastnosť je veľmi žiadúca pre budúceho administrátora systému, ktorý bude k webovej aplikácii pristupovať cez vzdialený prístup, pomocou terminálu, alebo príkazového riadka, a nebude mať k dispozícii editor so zložitým grafickým rozhraním.

K splneniu prehľadnosti šablón má výhodu prostý text. Poskytuje možnosť definovať pravidlo pre komentáre, čo JSON nepodporuje natívne. Formát JSON sa samozrejme dá upraviť a zadefinovať podľa požiadavku, ale tým by sme zablokovali jeden názov kľúča, keďže JSON pracuje s tzv. párom kľúč–hodnota [3]. Problém zablokovania názvu kľúča nastane pri jeho opätovnom zadefinovaní na rovnakej úrovni, ako je to znázornené vo výpise 3.1. Ďalšiou nevýhodou pravidiel súboru JSON je ten, že každý objekt musí byť ohraničený znakmi „{, }, [, ]“, čím pridáva priestor na chybu a pri dlhšej definícii väčšej tabuľky, značne predĺži šablónu. Táto nevýhoda má za následok horšiu prehľadnosť a nutnosť vývojára kontrolovať tieto znaky pri chybnom zápise.

```
{
    "comment": "Main birth entity",
    "main_obj":{
        "att_1": 1,
        "att_2": 2
    }
    "comment": "Person enties",
    "persons": [
        {
            "relation": "mother",
            "attributes": []
        },
        {
            "relation": "grand_mother",
            "attributes": []
        }
    ]
}
```

Výpis 3.1: Príklad nevalídneho objektu JSON.

Nevýhodou použitia prostého textu s vlastnými určenými pravidlami oproti formátu JSON, je ten, že JSON sú schopné prečítať a analyzovať rôzne vstavané knižnice pokročilých programovacích jazykov. Pre analýzu prostého textu bude potrebné vyvinúť vlastný syntaktický analyzátor.

V rámci tejto bakalárskej práce sme sa rozhodli navrhnúť šablóny vo formáte prostého textu a to z dôvodu prehľadnosti šablón.

## 3.4 Základný koncept

Splnením požiadaviek spomenutých v sekcii 3.2, šablóny implementujú mapovanie väzieb databázových tabuliek medzi sebou, a medzi databázou a stĺpcami súboru CSV (z angl. *Comma-separated values*), ktorý je očakávaným vstupom od užívateľa.

Formát CSV je určený k štruktúrovaniu dát do formátu tabuliek. Jeden úplný matričný záznam so všetkými možnými formátmi zodpovedá jednému riadku v súbore CSV ukončeným znakom nového riadka. Posledný riadok tabuľky znak nového riadka neobsahuje. Hodnoty buniek tabuľky sú zvyčajne oddelené čiarkou, ale môžu sa oddeľovať aj iným oddeľovačom, napr. bodkočiarkou (angl. *semicolon*). V prípade matričných dát je použitý, ako oddeľovač tabulátor [17].

Mapovanie medzi systémom a užívateľom funguje smerom dáta–databáza, ale aj databáza–dáta, čím je možné vykonať import a export matričných záznamov, s ktorými sa v danú chvíľu pracuje. Kvôli obojstrannému toku dát, boli navrhnuté dva hlavné typy šablón. Importné a exportné.

Importné šablóny slúžia pre kontrolu vstupných dát a obojstranné mapovanie jednotlivých častí matričného záznamu do príslušných atribútov databázovej tabuľky. Tieto šablóny garantujú vkladanie nových záznamov, jedného typu matričného záznamu, a aktualizáciu existujúcich dát v databáze systému DEMoS. Exportné šablóny slúžia k vygenerovaniu prázdneho CSV súboru, ktorý sa môže použiť k ručnému vyplneniu, a vygenerovaniu CSV súboru s dátami podľa špecifického filtra, daného užívateľom. Na export s dátami sa používa kombinácia exportných a importných šablón. Šablóny sa okrem importu a exportu používajú ako alternatíva k užívateľskému rozhraniu. Keďže šablóny definujú mapovanie, k zmene nie je potrebné zložité užívateľské rozhranie a stačí ich v textovom editore zmeniť.

Kvôli komplexnosti použitia šablón a popisu závislostí databázových entít, bol navrhnutý systém šablón, ktorý sa delí na šablóny popisujúce záhlavie tabuľky, ďalej na šablóny popisujúce hlavnú časť záznamu a na šablóny definujúce osoby, mená a pracovné postavenie osôb. Takto boli navrhnuté šablóny pre tri typy matričných záznamov a celkovo sa jedná o 15 šablón. Systém šablón je ešte rozšírený o konfiguračný súbor, ktorý obsahuje zoznam mien entít rozdelený do skupín. Rozhodnutie koľko a aké šablóny budú vytvorené, určovali tri faktory. Prvým faktorom bolo, o aké dáta sa jedná. Napríklad dátum narodenia sa viaže k udalosti narodenia, alebo titul otca dieťaťa sa viaže na konkrétnu osobu. Druhým faktorom bolo, ako často sa určitá časť opakuje z hľadiska vyššej abstrakcie. Tým sa myslí podobné definovanie, čím sú napríklad rôzne osoby vo vzťahu k udalosti. Tretím faktorom je pri akej akcii spracovania sa bude šablóna používať.

## 3.5 Popis štruktúry a syntaxe šablón

Pre udržanie jednoduchosti, prehľadnosti a štrukturizácie šablón, každá entita je zapísaná na každý riadok zvlášť, a teda napr. zadefinovanie databázovej tabuľky nemôže byť na rovnakom riadku. Syntax šablón dovoľuje aj vynechanie prázdnych riadkov ku kozmetickému oddeleniu jednotlivých entít šablóny.

Ako bolo naznačené, šablóny identifikujú niekoľko entít. Medzi ne patrí poznámka, popis skupiny atribútov, zadefinovanie cieľovej databázovej tabuľky, typ alebo kategória záznamu a reťazec popisujúci mapovanie atribútov, typ a dĺžka hodnoty, a formátovanie dát.

Šablónové entity „poznámka“ a „popis skupiny atribútov“ sa využívajú v exportných šablónach. Poznámka začína so znakom „#“, následovaná ľubovoľnou postupnosťou znakov a medzier, a zakončená je znakom nového riadku. Poznámka má len informatívnu funkciu

a na automatické vyhodnotenie nemá vplyv. Rovnako tak aj popis skupiny atribútov má informatívnu funkciu ale oproti poznámke, pri exporte je vložená do výstupného súboru. Skupina atribútov sa šablóne začína s „=#“. Ostatné riadky exportnej šablóny sú považované za názvy stĺpcov v CSV súbore.

Príklad exportnej šablóny je vo výpise 3.2, a správne definovanie poznámky, a databázovej tabuľky je vidieť v demonstračnom príklade vo výpise 3.3.

```
# Názvy stĺpcov v CSV súbore
=# Hlavná entita
spracovateľ/ka
Záznam hotový
Prosím o kontrolu
Archív
Fond
Signatura
Poradie skenu
```

Výpis 3.2: Príklad exportnej šablóny.

Všetky vyššie zmienené šablónové entity, okrem entity „skupiny atribútov“ sa používajú v importných šablónach. Syntax poznámky je rovnaká ako pri exporte. Zadefinovanie databázovej tabuľky začína s postupnosťou „>tbl>“, pokračujúcou s názvom tabuľky a ukončená je postupnosťou „<tbl“, ktorá sa nachádza v novom riadku. Detailnejší popis entity databázovej tabuľky je v kapitole 3.5.1. Podobnú logiku má aj kategória/typ časti z dát, len začínajúcou postupnosťou je „>cat>“ a ukončujúcou je „<cat“. Táto entita je podrobne rozobraná v kapitole 3.5.2. Poslednou entitou je mapovanie a popis atribútov vo forme reťazca. Reťazec je rozdelený do piatich častí pomocou dvojbodky „:“. Podrobnejšie vysvetlenie sa píše v neskoršej kapitole 3.5.3.

### 3.5.1 Definícia tabuľky v šablóne

V šablóne je tabuľka definovaná ako koreň stromu, pod ktorým sa nachádzajú 1 až N uzlov kategórií záznamu. V samotnej šablóne musí byť zadefinovaný aspoň jeden takýto strom, určujúci cieľovú tabuľku, tak ako je to znázornené vo výpise 3.3. Strom definujúci tabuľku nemôže byť podstromom iného stromu definujúceho inú tabuľku.

```
# Demonstračný príklad definície databázovej tabuľky a poznámky

>tbl>table_name_1

<tbl
```

Výpis 3.3: Definovanie tabuľky.

### 3.5.2 Kategórie

Kategórie vnášajú do šablón logický význam o akú časť dát z matričného záznamu sa jedná. Pre entitu kategórie platia podobné pravidlá ako pre entitu definujúcej tabuľku. Štruktúra kategórie je strom, ktorý obsahuje 1 až N uzlov, ktoré popisujú mapovanie atribútov. Každá entita kategórie musí byť podstromom entity definujúcej tabuľku a entity kategórii sa navzájom do seba nezanorujú, viď výpis 3.4.

```
# Demonstračný príklad definície kategórie entity a poznámky

>tbl>table_name_1
>cat>category_name_1

<cat
<tbl
```

Výpis 3.4: Definovanie kategórie.

### 3.5.3 Mapovanie hodnôt

Definícia mapovania jednej informácie v type matričného záznamu sa skladá z niekoľkých častí. Má tvar reťazca a začína názvom vstupného atribútu, pokračuje názvom atribútu databázovej tabuľky a ukončená je postupnosťou určujúcou typ, dĺžku a formát dát. Reťazec je listom stromu kategórie a je na poslednej úrovni stromu tabuľky. Vzorový príklad reťazca mapovania je vo výpise 3.5.

Prvky reťazca mapovania alebo aj n-tica mapovania obsahuje prvky, ktoré nadobúdajú znakovú hodnotu. Vývimkou sú prvok popisujúci maximálnu dĺžku prijímaných dát, ktorý má číselné hodnoty, a prvky bez podrobnejšieho opisu. Týmito prvkami sú názov atribútu CSV tabuľky, prvky bez potreby kontroly dĺžky a typ formátu pre dátový typ text, značený „TX“. Tento fenomén je v n-tici mapovania zapisovaný hviezdičkou „*“ a používajú ich už spomenutý dátový typ TX, ale aj formáty identifikátorov matričných udalostí, matričnej knihy, adresy a formát číselníka. Jednotlivým formátom sa bližšie venuje sekcia 3.5.5.

```
# Demonstračný príklad definície mapovania a poznámky

>tbl>table_name_1
>cat>category_name_1
csv attribute name 1:table_attribute_name:value_type:value_length:value_format
<cat
<tbl
```

Výpis 3.5: Definovanie mapovania atribútu.

### 3.5.4 Dátové typy mapovania

Dátový typ je druhým v poradí z prvkov n-tice mapovania. Vďaka jeho hodnote vie implementácia určiť stav konečného automatu spracovania kategórie, ktorého fungovanie je opísané v kapitole 4.2.4.

Hodnoty typov sú vkladané do objektu mapovania a časť z nich korešponduje s dátovými typmi databáze MySQL. Výnimkou sú typy cudzieho kľúča, konštantnej hodnoty a inkrementálnej hodnoty. Tieto tri typy nie sú plnohodnotné dátové typy, ale plnia rozhodovaciu rolu pri vkladania nových hodnôt do databáze systému. V nadchádzajúcich podkapitolách budú dátové typy priblížené.

#### Dátový typ cudzieho kľúča

Typ cudzieho kľúča „FK“ rozhoduje, že na mieste atribútu bude hodnota primárneho kľúča inej tabuľky. Tabuľku, ktorej primárny kľúč sa vyberie ako hodnota pre atribút typu „FK“, určuje nastavený druh formátu, ktorý sa nachádza v rovnakej n-tici mapovania. Druhy formátov budú vysvetlené v podkapitole 3.5.5.

### Dátový typ konštanty

Dátový typ konštantnej hodnoty slúži k testovacím účelom a k vloženiu hodnoty tzv. „silou“. Ako názov napovedá, tento typ určí k vloženiu konkrétnu hodnotu pre atribút n-tice mapovania, v ktorej sa nachádza. Príkladom môže byť hodnota *null*. Použitie tohto typu je hlavne určené pre atribúty, pri ktorých vieme hneď určiť hodnotu, alebo sú vo chvíli vkladania prázdne, a cieľový atribút databázovej tabuľky nemá východziu vlastnosť *DEFAULT NULL*[1], a bude neskôr upravovaný. Príklad nadefinovania reťazca mapovania s null konštantou je demonštrovaný vo výpise 3.6.

```
# Demonstračný príklad definície mapovania konštanty prázdnej hodnoty a poznámky

>tbl>table_name_1
>cat>category_name_1
csv attribute name 1:table_attribute_name:XX:NULL:XX
<cat
<tbl
```

Výpis 3.6: Mapovanie konštanty prázdnej hodnoty.

### Dátový typ inkrementálnej hodnoty

Inkrementálne hodnoty vnášajú možnosť vytvoriť niekoľko násobné väzobné prepojenie medzi dvoma tabuľkami a určiť ich poradie. Táto vlastnosť je použitá pri normalizácii mena osoby, ktoré sa skladá z niekoľkých krstných mien. Konkrétnym príkladom môže byť vloženie krstných mien osoby „Adam Lukáš Michal“. Mená sú ukladané do databázovej tabuľky „name“ a informácie o osobe v tabuľke „person“. V príklade s osobou s krstnými menami „Adam Lukáš Michal“ by sa vložili do prepojovacej tabuľky následovne:

| id | person_id | name_id | order |
|---|---|---|---|
| 1 | 11 | 21 | 1 |
| 2 | 11 | 22 | 2 |
| 3 | 11 | 23 | 3 |

Tabuľka 3.1: Väzobná tabuľka osoby a krstného mena.

### Dátový typ textu

Dátový typ textu sa v šablónach značí „TX“ a je rovnaký ako dátový typ databáze MySQL. Jedná sa o blok textu, ktorý má limit počtu znakov určený na 65 535 [23]. Pre tento vysoký limit sa pri dátovom type TX robí len kontrola či je prázdny. Príklad použitia dátového typu TX je ukázaný vo výpise 3.7.

```
# Demonstračný príklad definície mapovania bloku textovej hodnoty a poznámky

>tbl>table_name_1
>cat>category_name_1
csv attribute name 1:table_attribute_name:TX:*:*
<cat
<tbl
```

Výpis 3.7: Mapovanie textovej hodnoty.

[1] https://dev.mysql.com/doc/refman/8.0/en/data-type-defaults.html

#### Dátové typy číselných a znakových hodnôt

Význam číselných a reťazcových dátových typov je schopnosť kontrolovať dĺžku vkladaných dát ešte pred databázovým príkazom, a tým včas ošetriť reakciu na chybné dáta, a informovať užívateľa. Celý výčet dátových typov podporovaných spracovaním je vypísaný v nasledujúcej tabuľke 3.2.

| **Skupiny dátových typov** | **Hodnota** | |
|---|---|---|
| | **Šablóny** | **MySQL** |
| Číselné | TI | tiny int |
| | SI | small int |
| | ND | decimal |
| Reťazcové | C | char |
| | VC | varchar |
| | TX | text |
| | ENUM | enum |
| Cudzí kľúč | FK | - |
| Konštantná hodnota | XX | - |
| Inkrementálna hodnota | IN | - |

Tabuľka 3.2: Tabuľka dátových typov.

### 3.5.5 Formáty dát mapovania

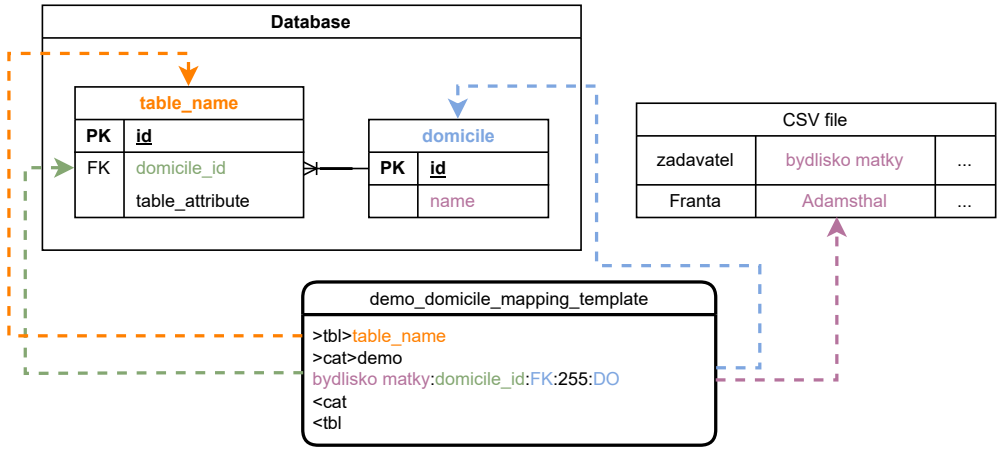
Formáty dát sú predposledným prvkom n-tice mapovania. Slúžia k rozšíreniu dátového typu šablóny, a určujú správny formát prijímaných dát, alebo spôsob generovania dát, ktoré nie sú súčasťou vstupných dát. Navrhnutá implementácia rozlišuje 36 rôznych formátov dát a sú prispôsobené špeciálne pre použitie systémom DEMoS. Jednoduché formáty sa starajú o správne naformátovanie dát pre vloženie do SQL príkazu, ako napr. formát znaku „c“, kedy sa po skontrolovaní počtu znakov, vytvorí hodnota rozšírená o apostrofy na začiatok a koniec hodnoty, tak by spĺňali syntax *insert* SQL príkazu. Zložitejšie formáty budú opísané v ďalších častiach a celý zoznam formátov dát sa nachádza v prílohe C.

#### Formát identifikátora

Hodnoty atribútov, ktoré sú pripojené z inej databázovej tabuľky majú vo všeobecnosti typ cudzí kľúč. Tento typ je rozšírený o jeden z formátov identifikátora. Každý formát má v implementácii staticky zadefinované, na akú tabuľku sa odkazuje.

Medzi najjednoduchšie z nich patria formáty užívateľa (*user, U*), vzťahu osoby (*person relation, PR*), povolania (*occupation, O*), sídlo/bydlisko (*domicile, DO*) a dôvod/spôsob úmrtia (*death cause, BD*). Jednoduchými formátmi sú preto, lebo v zázname sa identifikuje len jeden reťazec, uložený v jednom atribúte danej entity.

Príkladom môže byť mapovanie pre formát DO. Väzby medzi jednotlivými abstraktnými entitami je znázornený na obrázku 3.1. Formát DO určuje ako zdroj dát databázovú tabuľku *domicile*. Z tejto tabuľky sa bude vyberať primárny kľúč podľa neprázdnej hodnoty CSV atribútu stanovenou n-ticou mapovania, ktorá aj určuje maximálnu dĺžku, konkrétne 255 znakov. Primárny kľúč sa určí na vloženie do tabuľky atribútu rovnako podľa n-tice mapovania.




Obr. 3.1: Príklad mapovania formátu DO.

### Formát identifikátora matričnej udalosti a osoby

Formát matričného záznamu a osoby sú typy formátu, ktoré slúžia na prepojenie entít. Tieto formáty svoju hodnotu generujú počas spracovania dát a pomocou šablóny sa identifikuje, aká databázová entita potrebuje túto hodnotu pre vytvorenie väzby. Konkrétna hodnota je definovaná architektúrou databáze a jedná sa o primárny kľúč záznamu matričnej udalosti alebo osoby/osobnej udalosti. Matričné udalosti sa značia „B“ (*birth*, narodenie), „M“ (*marriage*, sobáš) a „BU“ (*burial*, pobreh). Formát osoby/osobnej udalosti sa značí veľkým písmenom „P“ (*person*). Príkladom osobnej udalosti môže byť sobáš už pri narodení dieťaťa. Nasledujúci výpis 3.8 poukazuje na spôsob zápisu mapovania formátov *B* a *P*.

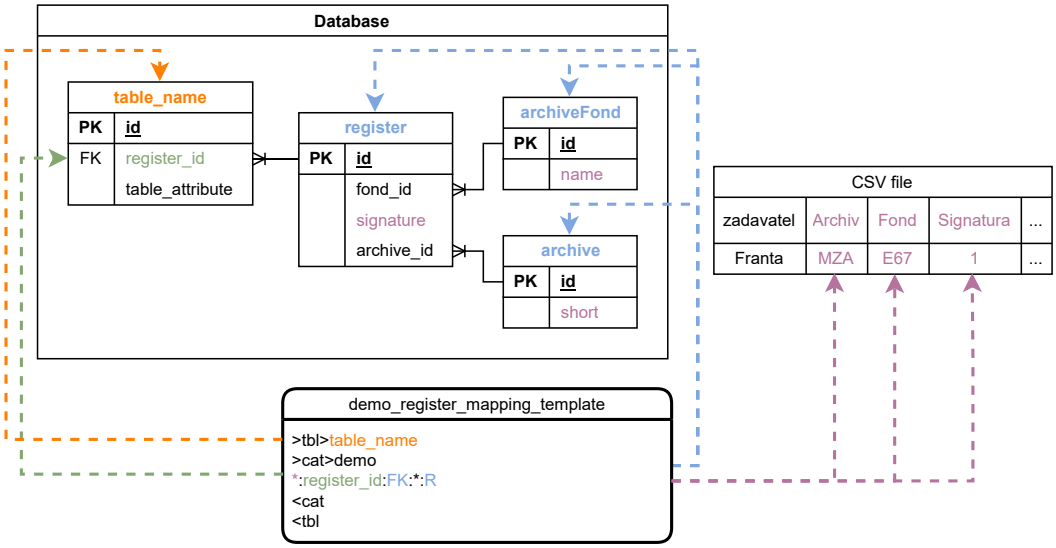
```
# Demonstračný príklad definície mapovania záznamu narodenia, a osoby, a poznámky

>tbl>table_name_1
>cat>category_name_1
*:table_birth_id:FK:*:B
*:table_person_id:FK:*:P
<cat
<tbl
```

Výpis 3.8: Mapovanie identifikátora matričnej udalosti a osoby.

### Formát identifikátora matriky

Jeden z dôležitých atribútov je identifikátor matrík. Tak ako je označovaná matričná kniha v kapitole 2.2, rovnako ju jednoznačne identifikuje aj databáza systému DEMoS. Určuje, do ktorej matričnej knihy, matričná udalosť patrí, a kde sa nachádza.

Formát matriky vytvára identifikátor z troch atribútov CSV tabuľky. Ide o atribút *Fond*, *Archiv* a *Signatura*. Navyše atribút matričnej knihy *Archiv* je odkazom do databázovej tabuľky „archive“ a atribút „Fond“ je odkazom do tabuľky *archiveFond*. Keďže sa identifikátor matriky dopočítava, nemá určený konkrétny stĺpec CSV tabuľky, názov CSV atribútu je označený hviezdičkou. Tento mechanizmus je braný ako nemenný, a preto je pevne daný v zdrojovom kóde, a šablóny obsahujú len jednu n-ticu, ktorej prvok formátu sa zapisuje „R“ (*register*). Demonstračný príklad použitia mapovania matriky je na nasledujúcom obrázku 3.2.

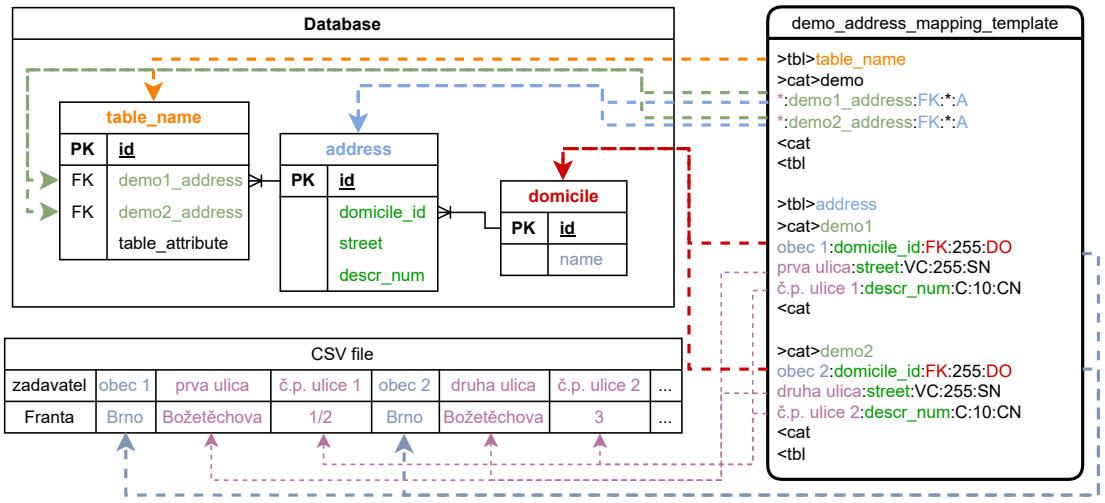



Obr. 3.2: Príklad mapovania formátu R.

### Formát identifikátora adresy

Formát adresy sa podobne ako formát matrík skladá z troch častí. Jedného odkazu do inej databázovej tabuľky a dvoch reťazcových atribútov. Odlišnosťou je zadefinovanie mapovania. Mapovanie je zadefinované v šablonách a jednotlivé časti sú zadefinované v zdrojovom kóde. Dôvodom pevného zadefinovania častí je ten, že sa nepredpokladá zmena definovania adries. Oproti formátu matričnej knihy, má formát adresy mapovanie definované v šablóne, z dôvodu možného opakovania sa. V šablóne je formtát adries značený písmenom „A“.

Samotné mapovanie adresy sa zadefinuje v ďalšom strome definovania tabuľky, znázornenom na obrázku 3.3. Previazanie atribútu hlavnej entity s entitou adresy vzniká, tak že sa názvu atribútu databázovej tabuľky nadefinuje meno so sufixom „_address“. Prefixom názvu tohto atribútu by malo byť jednoznačné označenie kategórie adresy. Prostredníctvom takto nadefinovaného názvu dokáže mať záznam matričnej udalosti viacero adries.



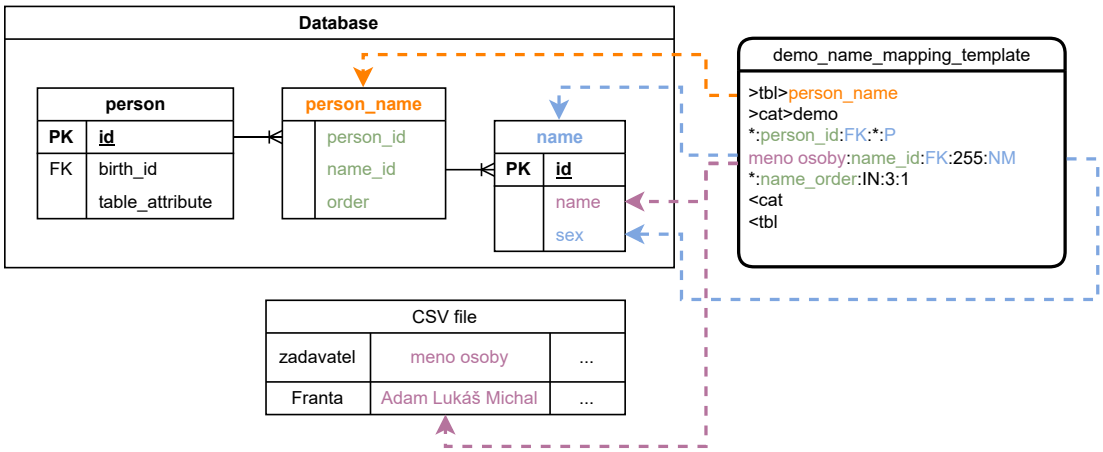

Obr. 3.3: Príklad mapovania formátu A.

### Formát identifikátora krstných mien a priezvisk

Formáty identifikátorov mien a priezvisk majú dokopy osem variant (formát mien má štyri varianty a formát priezvisk má tiež štyri). Varianty sa delia podľa rodu na mužské, ženské, neidentifikované a rodu určeného podľa pohlavia hlavnej osoby matričnej udalosti.

Značenie týchto formátov sa skladá z dvoch znakov. Krstné mená začínajú znakom „N“ a priezviská začínajú písmenom „S“. Ich druhý znak teda určuje variantu. Mužská varianta sa značí písmenom „M“ (*male*), ženská „F“ (*female*), neidentifikovaný rod „U“ (*undefined*) a varianta rodu na základe pohlavia určeného v zázname CSV tabuľky, sa označuje znakom „?“.

Tak ako bolo spomenuté v kapitole 3.5.4, entity osôb a mená osoby sú v architektúre databázy systému DEMoS prepojené väzobnou tabuľkou. Pre tento prípad sa môžu varianty formátu krstného mena rozšíriť o ďalšiu n-ticu mapovania s dátovým typom inkrementálnej hodnoty. Na nasledujúcom obrázku 3.4 je vykreslená logika mapovania medzi individuálnymi entitami mien a osôb.




Obr. 3.4: Príklad mapovania formátu NM rozšírený o typ IN.

### Formát číselníkov

Číselník, nazývaný aj ako výčtový typ (angl. *enumerated type*), je dátový typ mapovania, ktorý je rozšírený o päť formátov. Medzi tieto formáty patrí formát jazyka („LAN“, *language*), pohlavia („SX“, *sex*), legitimita („LEG“, *legitimate*), vzťahu („REL“, *relation*) a formát vierovyznania („REG“, *religion*).

Hodnoty týchto formátov boli určené architektúrou databáze a preto sú aj napevno nadefinované v implementácii. Celý výčet hodnôt je vypísaný v nasledujúcej tabuľke 3.3.

| Formát | Skratka | Definové hodnoty | Počet |
|---|---|---|---|
| jazyk | LAN | 'CZ', 'GE', 'LAT', 'SC', 'PL', 'SK', 'NONE' | 7 |
| pohlavie | SX | 'M', 'F', 'U', '[M]', '[F]' | 5 |
| legitimita | LEG | 'legitimate', 'illegitimate', 'U' | 3 |
| vzťah | REL | 'main', 'f', 'm', 'midwife', 'granted', 'f_f', 'f_m', 'f_m_f', 'f_m_m', 'm_f', 'm_m', 'm_m_f', 'm_m_m', 'gf_1', 'gf_2', 'gf_3', 'gf_4', 'gfrel_1', 'gfrel_2', 'gfrel_3', 'gfrel_4', 'husband', 'bapt_husband', 'mar_groom', 'mar_bride', 'mar_priest', 'mar_widower', 'mar_g_f', 'mar_g_m', 'mar_g_m_f', 'mar_g_m_m', 'mar_g_fost', 'mar_widow', 'mar_b_f', 'mar_b_m', 'mar_b_m_f', 'mar_b_m_m', 'mar_b_fost', 'mar_sv_1', 'mar_svrel_1', 'mar_sv_2', 'mar_svrel_2', 'mar_sv_3', 'mar_svrel_3', 'mar_sv_4', 'mar_svrel_4', 'mar_speaker', 'mar_stara', 'mar_bestman', 'mar_bridesmaid', 'bur_examinator', 'bur_keeper', 'bur_gravedigger', 'bur_main', 'bur_f', 'bur_m', 'bur_m_f', 'bur_m_m', 'bur_husband', 'bur_wife', 'bur_son', 'bur_daughter', 'bur_rel1' | 63 |
| vierovyznanie | REG | 'catholic', 'protestant', 'jew', 'none' | 4 |

Tabuľka 3.3: Tabuľka číselníkových formátov.

### Číselné, znakové a reťazcové formáty

Formáty číselných a znakových typov dát sú jednoduché formáty. Za zmienku stoja číselné formáty desatinného čísla „DE“ a znakové formáty dátumu „D“ a času „T“.

Formát DE definuje, že hodnota sa má skontrolovať na počet cifier pred a po desatinnej čiarke, definovaným prvkom n-tice mapovania dĺžky hodnoty. Prvok je zapísaný v tvare *xx.yy*, kde *xx* definuje počet cifier pred desatinnou čiarkou a *yy* za desatinnou čiarkou.

Formát D kontroluje tvar dátumu *DD.MM.YYYY* a transformuje ho na tvar *YYYY-MM-DD*, tak ako to je zadefinované systémom DEMoS. Pri chýbajúcich častiach dátumu sú časti nahradené znakom „?“. Formát času T kontroluje tvar zápisu časovej hodnoty na *HH:MM h*, prípadne *HH h*, a zapíše ich v rovnakom správnom tvare.

Ostatné formáty sú stručne popísané v tabuľke C.2. Praktické využitie časti číselných, znakových, reťazcových a ďalších formátov je znázornený vo výpise 3.9.

## 3.6 Šablóna hlavného záznamu úmrtia

V následujúcom výpise sa nachádza konkrétna šablóna matričnej udalosti úmrtia, ktorá bude použitá algoritmom spracovania dát. Jedná sa len o jednu šablónu a kompletný systém šablón sa nachádza v priloženom médiu.

```
>tbl>burial
>cat>main
*:register_id:FK:*:R
Pořadí scanu:scan:SI:5:N
Pořadí záznamu:pos:TI:3:N
rozložení na scanu (C, L, P):lay:C:1:LAY
Záznam hotov:fin:TI:1:BO
Jazyk záznamu:lang:ENUM:1:LAN
Pohlaví:sex:ENUM:1:SX
mrtvorozený:dead_born:TI:1:BN
datum zaopatření:viaticum_date:C:10:D
datum úmrtí:dead_date:C:10:D
datum pohřbení:burial_date:C:10:D
Přesný čas úmrtí:dead_time:TX:*:*
zaopatřen?:viaticum:TI:1:BN
místo pohřbení:burial_place:TX:*:*
místo úmrtí (pokud se liší od bydliště):death_place:TX:*:*
věk zemřelého roky:years:ND:5.2:DE
věk zemřelého měsíce:months:ND:5.2:DE
věk zemřelého týdny:weeks:ND:5.2:DE
věk zemřelého dny:days:ND:5.2:DE
věk zemřelého hodiny:hours:ND:5.2:DE
věk zemřelého minuty:minutes:ND:5.2:DE
příčina úmrtí:death_cause:FK:255:BD
ohledání A/N:examination:TI:1:BN
datum sňatku zemřelého:marriage_date:C:10:D
místo sňatku zemřelého:marriage_place:TX:*:*
pokřtěn:baptised:TI:1:BN
nemanželské:legitimate:ENUM:1:LEG
Počet let v manželství:marriage_years:ND:5.2:DE
poznámky:comment:TX:*:*
zpracovatel/ka:owner:FK:255:U
Prosím o kontrolu:check_req:TI:1:BO
*:death_address:FK:*:A
*:birth_address:FK:*:A
<cat
<tbl

>tbl>address
>cat>birth
obec narození zemřelého:domicile:FK:255:DO
ulice narození zemřelého:street:VC:255:SN
č. p. narození zemřelého:descr_num:C:10:CN
<cat
>cat>death
úmrtí obec:domicile:FK:255:DO
úmrtí ulice:street:VC:255:SN
úmrtí č. p.:descr_num:C:10:CN
<cat
<tbl
```

Výpis 3.9: Šablóna matričného záznamu úmrtia.

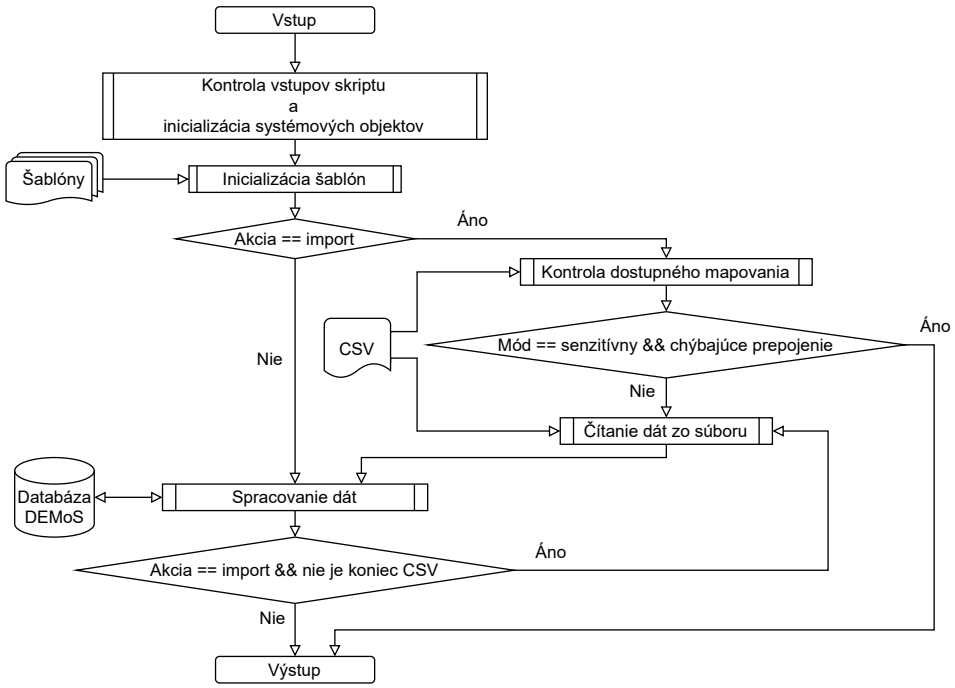
# Kapitola 4

# Implementácia modulu

Vývojom praktickej časti bakalárskej práce a zakomponovaním šablón, sa venuje nasledujúca kapitola. Kapitola sa bude zaoberať všeobecnému riešeniu spracovania dát a vysvetleniu entít a objektov, používaných aplikáciou. Všeobecná časť priblíži vo vyššej abstrakcii štruktúru aplikácie, a spoločné funkcie, a procesy importu, a exportu dát. Druhou v poradí je časť popisujúca mechanizmus importu, v ktorom vysvetlí vkladanie nových záznamov a editáciu existujúcich dát. Kapitola je zakončená algoritmom exportovania dát. Text každej z týchto časti bude rozšírený o vývojový diagram.

## 4.1 Základná štruktúra

Samotná implementácia je rozdelená do niekoľkých logických modulov, vyvinutých pomocou programovacieho jazyka Python vo verzii 3.10. Pohľad na architektúru aplikácie vo vyššej úrovni abstrakcie je vykreslený na obrázku 4.1 a kompletný diagram v prílohe B.1.




Obr. 4.1: Vývojový diagram implementovaného modulu.

Po spustení spracovania záznamov sa skontroluje validnosť vstupných prepínačov. Rovnako sa nastaví vykonávaná akcia, inicializujú a pripravia sa logovací súbor a pripojenie k databáze systému DEMoS. Existujúci systém využíva k uloženiu dát relačnú databázu MySQL. K pripojeniu a k práci s touto databázou bol využitý ovládač (angl. *driver*) MySQL Connector/Python[1]. Jednotlivé moduly sa bližšie priblížia v nasledujúcich sekciách.

### 4.1.1 Logovanie priebehu spracovania dát a štatistiky

Jeden z dôležitých systémových objektov je logovací súbor s príponou *.log*. Osobitné spracovania generujú vlastné logovacie súbory, čím sú informácie o behu izolované a lepšie vyhodnotiteľné. Mimo vlastného generovania logovacích súborov, sú súbory identifikované unikátnym názvom. Názov spočíva z dátumu vykonaného spracovania, čísla poradia v danom dni, ďalej typom akcie a na koniec hašom. Význam a výpočet haše v názve súboru je detailnejšie rozoberaný v kapitole 5.2.3.

Samotná analýza zozbieraných dát pomáhala k ladeniu a optimalizácii aplikácie pri vývoji. Okrem vývojových použití, logované dáta formujú štatistiky o dátach, ktoré boli analyzované a sprocesované. Tieto štatistiky obsahujú informácie o počte kontrolovaných matričných záznamov aplikáciou, o počte a mieste chýb, o počte databázových príkazov a úspešne vložených informáciách do databáze systému DEMoS. Účelom štatistík je vytvorenie odpovede na spustenie dávkového spracovania užívateľom, či už z webovej aplikácie, alebo z príkazového riadka. Vo výpise 4.1 je vybraná časť logovacích správ, kde formát jednej logovanej správy sa skladá v poradí:

- dátum a čas udalosti
- úroveň závažnosti
- názov udalosti
- telo správy

```
[2022-10-23 16:53:05] Debug: Cat: m Table suffix:
[2022-10-23 16:53:05] Debug: Category processing: m
[2022-10-23 16:53:05] Debug: Finished processing column name: birth_id
[2022-10-23 16:53:05] Debug: Finished processing column name: rel
[2022-10-23 16:53:05] Debug: Finished processing column name: title
[2022-10-23 16:53:05] Debug: Finished processing column name: sname
[2022-10-23 16:53:05] Debug: Finished processing column name: domicile
[2022-10-23 16:53:05] Debug: Finished processing column name: street
[2022-10-23 16:53:05] Debug: Finished processing column name: descr_num
[2022-10-23 16:53:05] Debug: Finished processing column name: religion
[2022-10-23 16:53:05] Debug: Finished processing column name: birth_date
[2022-10-23 16:53:05] Debug: Finished processing column name: waif
[2022-10-23 16:53:05] Debug: Empty row check: False, Insert flag check: True
[2022-10-23 16:53:05] Debug: Inserted row with ID: 335024
[2022-10-23 16:53:05] Debug: --Category done--
```

Výpis 4.1: Príklad obsahu logovacieho súboru.

[1] https://dev.mysql.com/doc/connector-python/en/

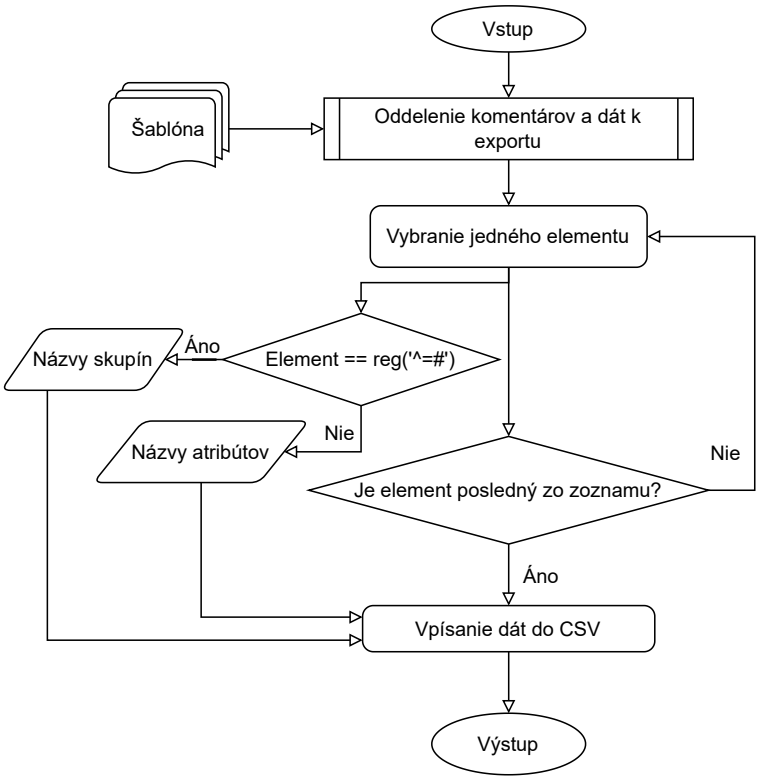
### 4.1.2 Načítanie definícií zo šablón

Po inicializácii súčastí a nástrojov aplikácie, sa podľa zadanej akcie pri spustení, načítajú prislúchajúce šablóny zo súborov. Relatívne cesty k šablónam sú definované v konfiguračnom súbore aplikácie.

Výsledkom načítania a spracovania môžu byť až tri entity. Dva reťazce, ktorý jeden z nich obsahuje hodnoty prvého riadku CSV súboru s názvami skupín stĺpcov a druhý udržiava názvy stĺpcov CSV súboru. Spolu reťazce tvoria záhlavie tabuľky. Treťou entitou je objekt tzv. slovník (angl. *dictionary*), obsahujúci zoznamy popisujúce mapovanie.

#### Reťazce záhlavia tabuľky

Reťazce sa používajú pri generovaní prázdnej CSV tabuľky, ale aj tabuľky s extrahovanými dátami z databáze. Algoritmus spracovania záhlavia tabuľky, zobrazený na obrázku 4.2, je rozdelený do dvoch častí. V prvej časti sa z načítanej šablóny oddelia komentáre a názvy elementov. V druhej časti sa názvy elementov rozdelia do finálnych reťazcov. Jednotlivé hodnoty názvov sú oddelené znakom tabulátoru. Finálny počet znakov tabulátoru je medzi reťazcami rovnaký, aby pri vložení do súboru CSV tvorili dva rovnako dlhé riadky.




Obr. 4.2: Diagram generovania záhlavia CSV tabuľky.

#### Objekt mapovania

Objekt slovníka je najkomplikovanejšou entitou aplikácie. Jeho funguje je založené na princípe kľúč–hodnota. V implementovanom riešení, kľúč reprezentuje názov cieľovej databázovej tabuľky a hodnotu zastupuje ďalší slovník kategórii. Kľúčom druhého slovníku je názov kategórie a hodnotou je pole n-tíc (angl. *array of tuples*).

V programovacom jazyku Python je n-tica postupnosťou, podobne ako reťazec, ale na rozdiel od reťazca, prvkami môžu byť hodnoty ľubovoľných dátových typov. Jedná sa o jeden z dátových typov jazyka, ktorý umožňuje ukladať kolekciu informácií [22].

Prvkami n-tice v navrhnutom riešení sú poradie stĺpca v súbore CSV, typ dát, dĺžka dát, formát dát a názov atribútu databázovej tabuľky. Ak prvok n-tice má hodnotu -1, tak sa hodnota nedá zo súboru CSV získať, a je buď generovaná, alebo sa nachádza v databáze. Vo výpise 4.2 je znázornený príklad n-tíc a štruktúra mapovania.

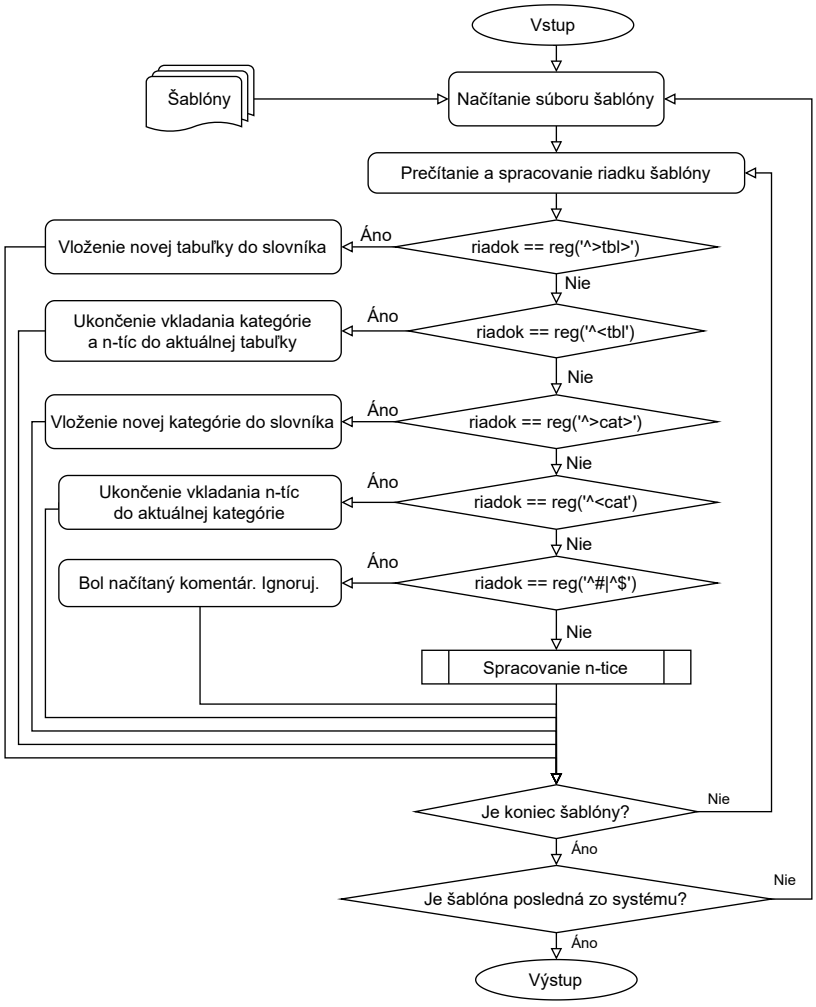
```
{
    'birth': {
        'main': [
            (-1, 'FK', -1, 'R', 'register_id'),
            (6, 'SI', 5, 'N', 'scan'),
            (8, 'TI', 3, 'N', 'pos'),
            (7, 'C', 1, 'LAY', 'lay')
        ]
    }, 'person': {
        'm': [
            (-1, 'FK', -1, 'B', 'birth_id'),
            (-1, 'ENUM', '1', 'REL', 'rel'),
            (83, 'VC', 255, 'SN', 'title')
        ], 'm_f': [
            (-1, 'FK', -1, 'B', 'birth_id'),
            (-1, 'ENUM', '1', 'REL', 'rel'),
            (94, 'VC', 255, 'SN', 'title')
        ]
    }, 'person_name': {
        'm-name': [
            (-1, 'FK', -1, 'P', 'person_id'),
            (84, 'FK', '*', 'NF', 'name_id'),
            (-1, 'IN', 3, 1, 'name_order')
        ], 'm_f-name': [
            (-1, 'FK', -1, 'P', 'person_id'),
            (95, 'FK', '*', 'NM', 'name_id'),
            (-1, 'IN', 3, 1, 'name_order')
        ]
    }
}
```

Výpis 4.2: Príklad obsahu objektu mapovania.

### 4.1.3 Algoritmus vytvorenia objektu mapovania

Pri spracovaní dát použitím akcie import a export s dátami, implementácia sprocesuje importné šablóny, spomenuté v kapitole 3.4. Algoritmus postupne prechádza systém šablón súbor po súbore, tak ako to je vykreslené na obrázku 4.3 vývojového diagramu.

Samotný súbor spracúva po riadkoch a tvorí entitu mapovania. Dodržiavaním zadefinovanej syntaxe tabuľky z kapitoly 3.5.1, vloží názov do slovníku mapovania a postupne vkladá do slovníku kategórií ich názvy až kým výpočet nenarazí na reťazec konca tabuľky „<tbl“. Pred ukončením vyhodnocovania tabuľky, podobný spôsobom spracuje aj zanorený slovník kategórií, do ktorej vkladá pole n-tíc.



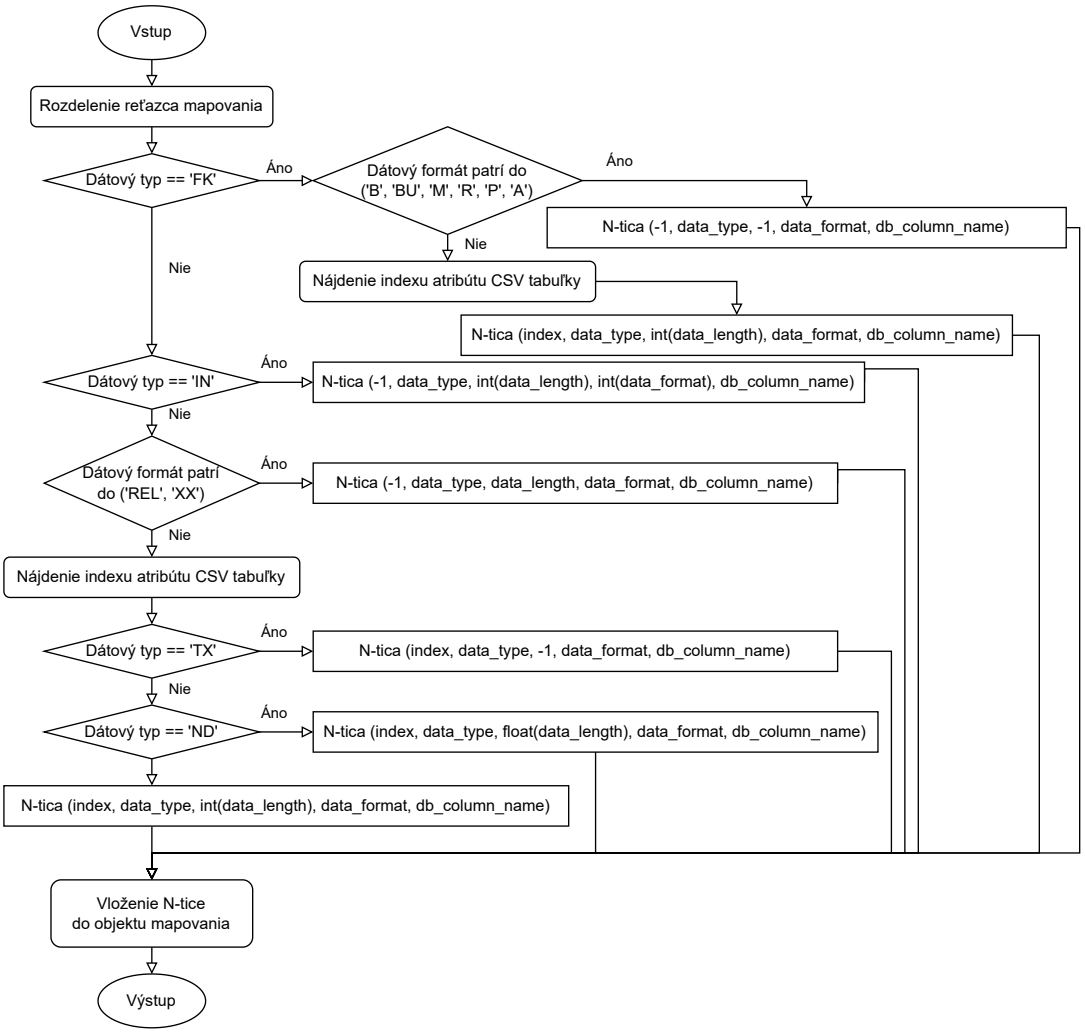

Obr. 4.3: Vývojový diagram generovania objektu mapovania.

Rozdelením reťazca sú n-tice vytvárané, tak ako to je definované v kapitole 3.5.3. Po rozdelení reťazca mapovania na jednotlivé prvky, algoritmus vytvárania objektu mapovania prejde do druhého stavu spracovania, v ktorom spracuje prvky a vloží ich na správne miesta novej n-tice. Tento mechanizmus je znázornený na diagrame 4.4.

Generovanie n-tice je podmienené stavovým automatom, ktorého stavy určujú prvky reťazca mapovania. Stavy sa delia na stav cudzieho kľúča, inkrementálnej hodnoty, ďalej formát vzťahu a prázdnej hodnoty, a na koniec ostatných typov formátov. Stav cudzieho kľúča sa ďalej delí na formáty matričných udalostí, osoby, adresy a ostatné formáty cudzieho kľúča.

Pri generovaní n-tice sa implementácia snaží nájsť umiestnenie tzv. index atribútu v záhlaví CSV tabuľky. Index môže nadobúdať hodnoty celých čísel v intervale $\langle -1; +\infty)$. Ako už bolo spomenuté, index s hodnotou -1 je ukazovateľ na umiestnenie, ktorý hovorí, že sa vkladaná hodnota v tabuľke nenachádza a musí sa dopočítať, alebo je vkladaná konštanta. Príkladom môže byť mapovanie pre kategóriu „m“ tabuľky „person“, zobrazené vo výpise 4.2. Jedna z n-tíc obsahuje prvok „ENUM“. Mapovanie tohto atribútu nemá určené, z ktorého indexu záhlavia by mala byť hodnota prebratá. V tomto konkrétnom prípade, hodnota atribútu popisovaného n-ticou, sa prevezme z názvu kategórie pri vkladaní do da-

tabáze, a teda to bude hodnota „m“. Pravidlo, ktoré určí zdroj hodnôt, definuje formát dát z n-tice.




Obr. 4.4: Vývojový diagram generovania jednej n-tice mapovania.

Dokončením potrebných inicializácii a prepojení, následuje samotné spracovanie dát. Importu dát sa podrobnejšie venuje podkapitola 4.2 a výpočtu exportu podkapitola 4.3.

## 4.2 Algoritmus importu záznamov

Následujúce sekcie tejto podkapitoly budú vysvetlovať logické časti procesu vkladania nových a aktualizovanie existujúcich dát. Ako prvé sa vysvetlí kontrola mapovania dostupných atribútov v tabuľke CSV a v šablónach, ktoré je možné vykonať úplne a neúplne. Ďalej sa priblíži význam kontroly povinných atribútov. Následovať nato budú sekcia vyhodnotenia o aký druh vloženia sa jedná a postupnosť sekcií, ktoré priblížia spracovanie hlavnej časti matričnej udalosti a osobám vo vzťahu k udalosti.

### 4.2.1 Kontrola mapovania

Kontrola dostupného mapovania sa vykonáva pri každom spracovaní typu vkladania dát, a zisťuje, či všetky atribúty z n-tice mapovania majú pár s nejakým atribútom záhlavia v CSV tabuľke. Formálne sa jedná o rovnosť dvoch množín, ktorá sa zapíše:

$$A = B \Leftrightarrow (\forall_x A : x \in B \wedge \forall_x B : x \in A)$$

Algoritmus dokáže fungovať aj v prípade, že mapovanie nie je úplne. Kritérium pre neúplné mapovanie je, aby každý atribút zo šablóny mal prepojenie s nejakým stĺpcom v tabuľke CSV. Prepojením sa chápe rovnosť dvoch reťazcov. Matematicky sa jedná o podmnožinu inej množiny a teda všetky prvky n-tíc, popisujúce názvy atribútov záhlavia CSV tabuľky, sú podmnožinou množiny atribútov vstupného CSV súboru. Formálny matematický zápis je následovný:

$$A \subset B \Leftrightarrow (A \subseteq B \wedge A \neq B)$$

Na výsledok kontroly implementácia reaguje v dvoch módoch. Senzitívnom a nesezitívnom. Pri senzitívnom je kontrola úspešná pri úplnom mapovaní a nesezitívny mód považuje za úspešné aj neúplné mapovanie.

### 4.2.2 Kontrola dostupnosti povinných údajov

Predposledná kontrola pred vkladaním záznamu do systému spočíva v tom, že sa z CSV riadku s dátami určenými k vloženiu, získa meno užívateľa a identifikácia knihy alebo listiny daného matričného záznamu. Tieto povinné hodnoty nie sú definované šablónami a ide o kritérium systému DEMoS, zmienenú v kapitole 2.6.

Účelom kontroly je predísť vkladaniu záznamov, ktoré nie sú priradené k žiadnemu matričnému celku a predísť stráte informácie, kto vložil dané dáta do systému. Užívateľ sa kontroluje podľa prezývky.

### 4.2.3 Rozlíšenie druhu spracovania

Vstupnými dátami vkladania môžu byť nové matričné záznamy, alebo upravené matričné záznamy existujúce v genealogickej databáze DEMoS. Rozlíšenie, o aký druh SQL príkazu sa bude jednať, algoritmus rozpozná pomocou jednoznačného identifikátoru matričnej knihy, alebo listity, a fyzického umiestnenia v danom fyzikom médiu.

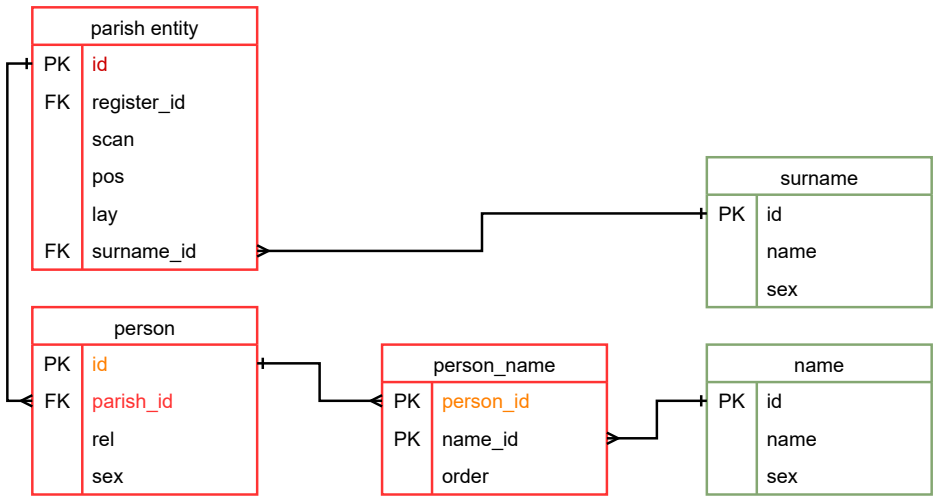
Jednoznačné identifikovanie fyzického média určujú názov, alebo skratka archívu, ďalej názov fondu, a signatúra popisujúca médium. Spomenuté identifikovanie je v súlade jedna k jednej s archívnym identifikovaním záznamov, popísaných v kapitole 2.2.

Identifikátor umiestnenia sa skladá z informácie poradia skenu vykonanom na fyzickom médiu, pokračuje s poradím daného záznamu na skene a rozložením na skenu. Rozloženie môže hovoriť, či daný záznam je umiestnený vľavo, vpravo, alebo v strede skenu.

Požadovaným chovaním aktualizácie implementovaného modulu, je prepísanie všetkých prislúchajúcich atribútov hlavných entít. Preto pri potvrdení, že vkladané dáta už v databáze existujú, implementácia vykoná SQL príkaz *DELETE*. Vďaka vlastnosti architektúry databáze, ktorá má nastavené cudzie kľúče na tabuľky *birth*, *marriage* a *burial* s *ON DELETE CASCADE*, je možné jednoducho matričný záznam vymazať a následne znovu vložiť. Keďže hlavnými entitami sú tabuľky birth/marriage/burial, person a prepojovacie tabuľky rôzneho druhu, kaskádové zmazanie záznamu hlavnej udalosti, zmaže aj ostatné

hlavné záznamy tabuliek, viď obrázok 4.5. Dosah vymazania záznamov je označený červenou farbou. Ostatné entity ako krstné mená, priezviská sú považované za akési slovníky, alebo dynamické číselníky, a ich zmazanie nie je požadované. Tento príkaz je veľmi citlivý, a preto sa databázová udalosť zapíše, aj s hodnotami do logovacieho súboru pre prípadné obnovenie dát.

Pri vyhodnotení, že sa jedná o nový záznam alebo po dokončení zmazania matričného záznamu, sa uskutoční spracovanie dát. Odlišnosťou od vkladania úplne novej informácie a aktualizovania dát, je tá, že pri aktualizovaní sa použije starý identifikátor hlavnej udalosti. Novému záznamu hlavnej udalosti sa automaticky vygeneruje nový identifikátor. Novým a aktualizovaným záznamom osôb a osobných udalostí sa tiež vygeneruje nový identifikátor.



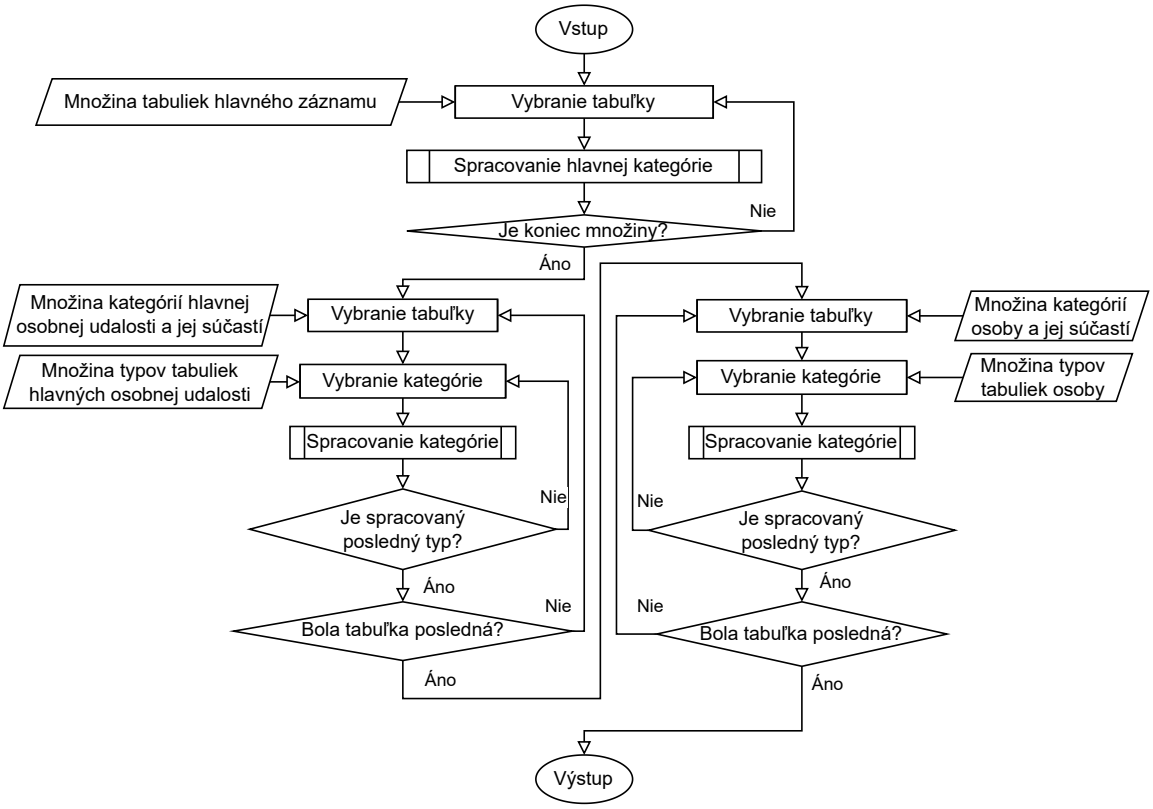

Obr. 4.5: Grafické znázornenie dosahu zmazania matričného záznamu.

### 4.2.4 Spracovanie záznamu

Spracovanie jedného záznamu sa deje po jednotlivých častiach tzv. kategóriách. Priebeh spracovania je znázornený na diagrame 4.6. Ako prvá kategória sa spracuje hlavná udalosť matričného záznamu, ktorou môže byť, buď záznam narodenia, sobášu, alebo úmrtia. Po dokončení spracovania tejto časti dát, sa prejde na vyhodnotenie všetkých kategórií, ktoré súvisia s hlavnou kategóriou. Ku príkladu sa môže dať hlavný záznam narodenia a birmovného mena, ktoré sa v historických listinách písali k záznamu narodenia.

Spracovaním hlavnej osobnej udalosti sa pokračuje ku kategóriám osôb a ich súčastí, ktoré majú vzťah k hlavnej entite. Najprv sa spracuje kategória samotnej osoby a následne na to, sa vyhodnotia mená a pracovné povolania. Informácie o menách, a pracovných pôsobení, či titulov, sú kvôli normalizácii dát v oddelených databázových tabuľkách, preto aj majú vlastné kategórie.

Niektoré kombinácie osoby vo vzťahu k udalosti a pracovnému povolaniu nedávajú zmysel, preto implementácia umožňuje vytvoriť výnimku týchto kombinácii. Zadefinovanie špeciálnych kombinácii, ktoré nemajú pre matričný záznam význam, sú definované implementáciou a nie šablónou. Zoznam výnimiek sa nachádza v konfiguračnom súbore implementovaného modulu. Príkladom opisovanej výnimky je napríklad osoba, ktorá pochovala zomrelého alebo osoba, ktorá vykonala svadobný obrad. Pri oboch osobách je z názvu zrejmé, aké povolanie vykonávali.




Obr. 4.6: Vývojový diagram spracovanie matričného záznamu.

### 4.2.5 Spracovanie kategórie

Mechanizmus spracovania dát sa vykonáva pre každú kategóriu tým, že sa zo slovníku mapovania postupne vyberie jedna n-tica z množiny. Vybraná n-tica popisuje potrebné prepojenia, podľa názvu tabuľky a názvu kategórie, ktoré boli doňho vložné za pomoci algoritmu popísanom v kapitole 4.1.2. Mechanizmus spracovania kategórie je v zjednodušenej podobe vykreslený diagramom na obrázku 4.7.

Cyklicky sa vyhodnotia n-tice pomocou stavového automatu, ktorého hodnoty určuje druhý prvok n-tice. Druhým prvkom je identifikátor dátového typu. V tomto kroku sa skontroluje dĺžka dát, po ktorej následuje pomocou ďalších stavov, kontrola formátu dát. Pre kontrolu jednotlivých formátov bol vytvorený samostatný zdrojový súbor. Dátové typy TX, IN a XX rozširujúce formáty nemajú, z toho dôvodu je ich spracovanie v zdrojovom súbore spracovania záznamu.

Po dokončení spracovania celej jednej kategórie, algoritmus vyhodnotí či boli nájdené nejaké chyby v dátach. Pri úspešnom spracovaní, dochádza ku kontrole či nie je aktuálne spracovávaná časť matričného záznamu prázdna, aby sa predišlo vkladaniu redundantných dát. Redundantnými dátami rozumieme informácie, napríklad neznámej osoby otca dieťaťa, ktoré v bunkách riadku CSV tabuľky neobsahujú žiadne údaje.

Overením neprázdnych dát, algoritmus vygeneruje príslušné SQL príkazy, definované v kapitole 4.2.3. Pri úspešnom spracovaní celého riadku z CSV tabuľky, algoritmus potvrdí databázové zmeny SQL príkazom *COMMIT*. Opačnom prípade použije príkaz *ROLLBACK* a tým vráti záznam do pôvodného stavu, čím vráti do tabuliek aj vymazané dáta pri aktualizácii existujúceho záznamu.


Obr. 4.7: Vývojový diagram spracovanie kategórie matričného záznamu.

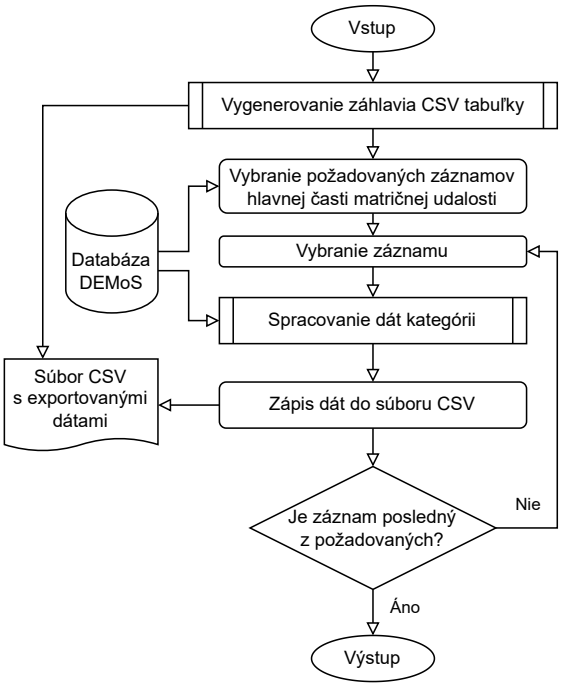
## 4.3 Algoritmus exportu záznamov

Export funguje v dvoch konfiguráciách. Prvá už bola popísaná v sekcii 4.1.2 a vytvára súbor formátu CSV obsahujúci záhlavie tabuľky. Výstup tohto druhu exportu slúži k ručnému vytvoreniu dávky dát k následnému spracovaniu a vloženiu do systému DEMoS.

Druhá konfigurácia umožňuje rovnako ako prvá časť vygenerovať súbor CSV so záhlavým tabuľky, ale s tým rozdielom, že do tela tabuľky extrahuje uložené dáta z databáze. Množinu matričných dát identifikuje pole unikátnych identifikátorov hlavnej udalosti, ktorá je povinným vstupným parametrom spustenia exportu dát. Tento algoritmus je v jednoduchšej podobe vizualizovaný diagramom na obrázku 4.8.

Algoritmus exportu dát využíva, aj ako šablóny záhlavia tabuľky, tak šablóny mapovania, preto jeho logické časti majú podobu aspektov vo výpočte. Algoritmus po inicializácii potrebných aplikačných entít, vytvorí súbor CSV, do ktorého bude postupne vkladať extrahované dáta. Mechanizmus generovania exportných súborov CSV je bližšie popísaný v kapitole 4.3.1. Výpočet pokračuje ďalej vytvorením SQL príkazu na výber dát hlavných udalostí, kde sa zakomponujú na vstup aplikácie vložené unikátne identifikátory a názvy atribútov hlavnej udalosti. Pre každý výsledok SQL príkazu sa postupuje podobne ako pri importe dát. Najprv sa zadeklaruje a inicializuje objekt typu pole o veľkosti počtu atribútov CSV tabuľky, ktoré bude slúžiť ako dočasné úložisko extrahovaných dát. Ďalej sa spracuje hlavná udalosť, a potom postupne entity naviazané na hlavnú udalosť, a jednotlivé osoby vo vzťahu k udalosti. Spracovanie a vkladanie jednotlivých atribútov entít bude pri-

blížený v kapitole 4.3.2. Posledným krokom exportu je z extrahovaných dát vytvoriť riadok tabuľky a vložiť ho do vygenerovaného súboru CSV.




Obr. 4.8: Vývojový diagram exportu dát.

### 4.3.1 Generovanie súboru

Výstupom exportu je súbor vo formáte CSV. Súbor sa ukladá podľa druhu exportu, buď do priečinku *empty_csv*, alebo do *data_csv*, ktoré sa nachádzajú v priečinku *exports*.

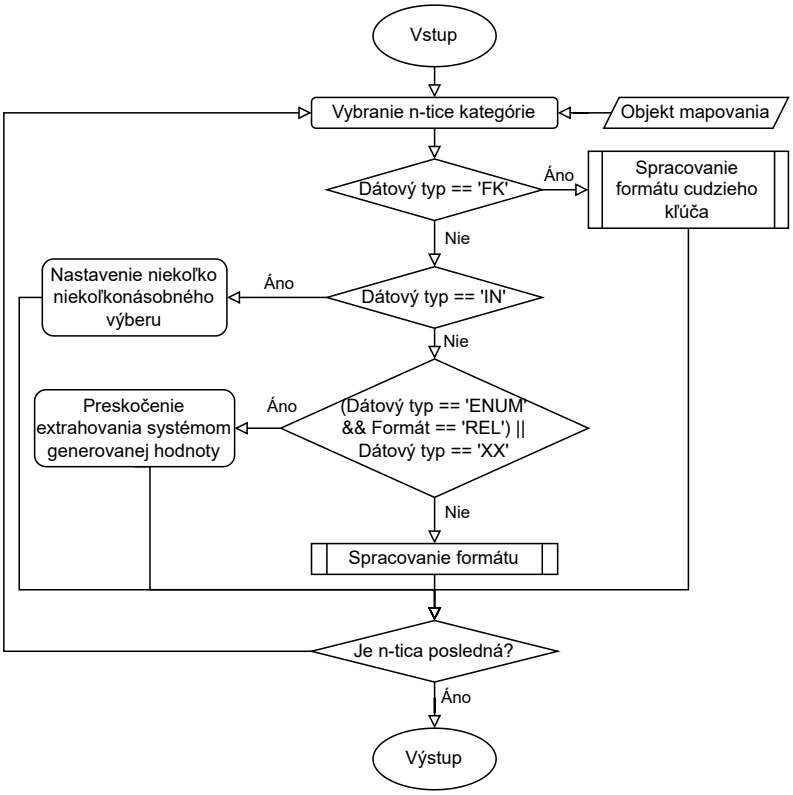
Názov vygenerovaného súboru pre schopnosť odlíšenia, sa musí skladať z unikátneho názvu. Názov sa skladá teda z dátumu a času generovania, čísla poradia v danom dni, typu matričnej tabuľky a hashe vygenerovanej webovou aplikáciou.

Pre šetrenie uložiskového miesta na serveri, bol do súboru *Makefile* naimplementovaný príkaz na vymazanie všetkých alebo časti exportovaných súborov.

### 4.3.2 Skladanie dát do CSV

Algoritmus skladania dát funguje podobne ako spracovanie n-tice mapovania popísané v kapitole 4.2.5. Každej kategórii sa postupne extrahujú dáta príslušnej časti z n-tice, ktorá sa nachádza v slovníku mapovania pod názvom kategórie. Mechamizmus je založený na stavom automate, ktorého stavy určuje hodnota dátového typu. Logika mechanizmu je zobrazená na diagrame 4.9.

Pri procese importu bola vykonaná na určitýh typoch dát transformácia, čím sa zjednotil formát dát. Spätná transformácia dát pri exporte sa nevykonáva. Z toho dôvodu bola rozšírená kontrola vstupných dát o už transformované dáta. Konkrétne sa jedná o formáty čísleníkových dátových typov (LAN, SX, LEG, REG), formáty pravdivostnej hodnoty (B0, B1, BN) a o formát dátumu (D).




Obr. 4.9: Vývojový diagram spracovania kategórie pri exporte dát.

## 4.4 Príklad rozšírenia šablón

Nasledujúca podkapitola bude obsahovať simulované praktické ukážky rozšírenia systému šablón. Prvej časti sa ukáže jednoduché pridanie nových atribútov entity. V druhej časti sa demonštruje rozšírenie novej tabuľky a v poslednej časti sa ukáže spôsob aktualizácie architektúry databáze o tabuľku prepojenú cez väzobnú tabuľku.

### 4.4.1 Pridanie nových atribútov tabuľky

Nasledujúci príklad ukáže reakciu na pridanie atribútov do databázovej tabuľky *person*. Tabuľka pre tento príklad bude rozšírená pomocou SQL príkazy znázorného vo výpise 4.3. Upravovaným typom spracovania bude matričný záznam narodenia.

Rozšírené atribúty budú dva. Jeden znakový, ktorý bude mať maximálne štyri znaky, v CSV tabuľke sa bude pre osobu matky nachádzať v stĺpci s názvom „Prezývka matky“ a v databázovej tabuľke sa bude nachádzať v atribúte „alias“. Druhým atribútom bude poznámka k osobe matky. V tabuľke CSV sa bude nachádzať pod stĺpcom „Poznámka k matke“ a v databázovej tabuľke v atribúte „note“. Potrebné rozšírenie šablóny osoby matky sa nachádza vo výpise 4.4.

```
1 ALTER TABLE person
2 ADD alias CHAR(4) NULL,
3 ADD note TEXT NULL;
```

Výpis 4.3: Príkaz SQL rozšírujúci tabuľku *person*.

```
>cat>m
*:birth_id:FK:*:B
*:rel:ENUM:1:REL
Titul matky:title:VC:255:SN
Příjmení matky:sname:FK:255:SF
Obec matky:domicile:FK:255:DO
ulice matky:street:VC:255:SN
č. p. matky:descr_num:C:10:CN
Vyznání matky:religion:ENUM:1:REG
datum narození matky:birth_date:C:10:D
Nalezenec matka:waif:TI:1:BN

Prezývka matky:alias:C:4:CN
Poznámka k matke:note:TX:*:*
<cat
```

Výpis 4.4: Časť šablóny *birth-peron.def* po úprave.

### 4.4.2 Pridanie novej tabuľky

Nasledujúca praktická ukážka simuluje vytvorenie novej tabuľky, ktorá sa bude odkazovať na hlavnú udalosť úmrtia. Tabuľka bude uchovávať informácie kde (ústav a obec), od kedy, do kedy bol nebohý pred vložením do hrobu uložený.

V tomto prípade je potrebné meno novej tabuľky vložiť do konfiguračného súboru *config.py*, do slovníku *definition.burial.main_related_tables*. Rozšírujúci SQL príkaz je zobrazený vo výpise 4.5 a mapovanie novej tabuľky, ktorá by mala byť vložená do šablóny, je zobrazená vo výpise 4.6.

```
CREATE TABLE tacr.burial_before (
    id int auto_increment primary key,
    burial_id int unsigned not null,
    where varchar(10) null,
    from char(10) null,
    to char(10) null,
    domicile  mediumint unsigned null,

constraint burial_before_infk_1
    foreign key (burial_id) references tacr.burial (id)
        on delete cascade,
constraint burial_before_ibfk_2
    foreign key (domicile) references tacr.domicile (id)
        on delete cascade
);
```

Výpis 4.5: Príkaz SQL vytvárajúci novú tabuľku.

```
>tbl>burial_before
>cat>main
*:burial_id:FK:*:BU
Uložený kde:where:VC:255:SN
Uložený od:from:C:10:D
Uložený do:to:C:10:D
Uložený obec:domicile:FK:255:DO
<cat
<tbl
```

Výpis 4.6: Definícia mapovania novej tabuľky po úprave.

### 4.4.3 Pridanie novej entity a väzobnej tabuľky

Posledná ukážka priblíži zmenu architektúry systému, kedy sa vytvorí nová tabuľka (podobná menám a povolaniam), ktorá bude slúžiť na uchovanie informácie, aké dary dostalo novonarodené dieťa. Okrem názvu daru, bude tabuľka obsahovať krstné mená darcu, priezvisko darcu a cenu daru. Darcov môže byť maximálne pre tento príklad dvaja.

Kedže sa idú vytvárať nová tabuľka, a k nej väzobná tabuľka medzi menom dárcom a darom, je potrebné do konfiguračného súboru vložiť názov novej tabuľky, a sufix názvu väzobnej tabuľky. Vo výpisoch 4.9, 4.8 a 4.7 sú zobrazené rozšírené informácie jednotlivých entít.

```
>tbl>birth_gift
>cat>gift_1
*:birth_id:FK:*:B
*:num:XX:1:XX
Nazov daru 1:name:C:50:CN
Priezvisko darcu 1:sname:FK:255:SU
Cena daru 1:price:ND:5.2:DE
<cat

>cat>gift_2
*:birth_id:FK:*:B
*:num:XX:2:XX
Nazov daru 2:name:C:50:CN
Priezvisko darcu 2:sname:FK:255:SU
Cena daru 2:price:ND:5.2:DE
<cat
<tbl

>tbl>birth_gift_name
>cat>gift_1_name
*:gift_id:FK:*:P
Meno darcu 1:name_id:FK:255:NU
*:name_order:IN:3:1
<cat

>cat>gift_2_name
*:gift_id:FK:*:P
Meno darcu 2:name_id:FK:255:NU
*:name_order:IN:3:1
<cat
<tbl
```

Výpis 4.7: Definícia mapovania novej tabuľky a väzby na meno.

```
CREATE TABLE tacr.birth_gift (
    id          int                 auto_increment primary key,
    birth_id    int unsigned        not null,
    num         tinyint unsigned    null
    name        varchar(10)         null,
    sname       mediumint unsigned  null,
    price       decimal(5, 2)       null,

constraint burial_before_infk_1
    foreign key (birth_id) references tacr.birth (id)
        on delete cascade,
constraint burial_before_ibfk_2
    foreign key (sname) references tacr.surname (id)
            on delete cascade
)

CREATE TABLE tacr.birth_gift_name (
    gift_id    int unsigned       not null,
    name_id    mediumint unsigned not null,
    name_order tinyint unsigned   not null,

    primary key (gift_id, name_id),
    constraint birthMarriage_name_ibfk_1
        foreign key (gift_id) references tacr.birth_gift (id)
            on delete cascade,
    constraint birthMarriage_name_ibfk_2
        foreign key (name_id) references tacr.name (id)
            on delete cascade
);
```

Výpis 4.8: Príkaz SQL vytvárajúci novú tabuľku a väzobnú tabuľku.

```
definition = { "birth": {
        "main_related_fk_names": ["gift_id"],
        "main_related_categories": ["gift_1", "gift_2"],
        "main_related_suffixes": ["", "_name"],
    }
}
```

Výpis 4.9: Časť slovníku kofiguračného súboru potrebné k pridaniu novej entity a väzby.

# Kapitola 5

# Pripojenie na webovú aplikáciu

Ďalším krokom vývoja navrhnutého riešenia je jeho zaintegrovanie do webového rozhrania systému DEMoS. Tomuto procesu sa venuje nadchádzajúca kapitola. Obsahom podkapitol bude popis fungovania webového serveru a nevyhnutné inštalačné procedúry, ktoré umožnia budúce fungovanie rozšírenia aplikácie. Nasledujúcim obsahom bude vysvetlenie architektúry webovej aplikácie a úpravy, ktoré boli vykonané pri naviazaní webového rozhrania s navrhnutou implementáciou spracúvajúcou genealogické dáta.

## 5.1 Webový server

Prostredia systému DEMoS bežia na dvoch odlišných distribúciách operačného systému Linux, Ubuntu a CentOS. Jednou hlavnou odlišnosťou operačných systémov je politika práv.

Na distribúcii Ubuntu stačí, pre správne používanie a fungovanie naimplmentovaného modulu, aby sa nastavili práva na manipuláciu a spúšťanie súborov. Na druhej strane, distribúcia CentOS má rožšírené jadro operačného systému o tzv. SELinux (angl. *Security-Enhanced Linux*). Toto rozšírenie prináša možnosť prideľovať oprávnenia k vykonaniu určitého úkonu na úrovni jednotlivých procesov, užívateľov atď. [24].

Medzi ďalší problém patrí nepripravenosť aplikačných balíčkov a nutnosť ich nainštalovať. Pre tieto úskalia bol subor *Makefile* rozšírený o potrebné inštalačné príkazy potrebných balíčkov a taktiež aj príkazy, ktoré nastavujú práva podľa požiadaviek operačného systému.

O fungovanie webového serveru sa stará webový server Apache[1]. Implementácia počíta s tým, že webová aplikácia má spravne nakonfigurované odkazy a oprávnenia v konfiguračných súboroch aplikácie Apache.

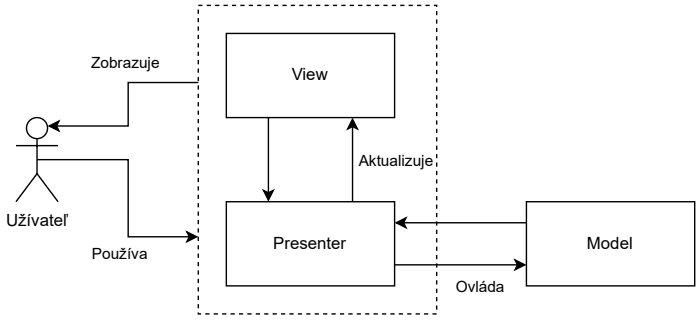
## 5.2 Webová aplikácia

Webová aplikácia systému DEMoS je naprogramovaná v programovacom jazyku PHP. Aplikácia využíva open source framework Nette[2] a ďalšie komponenty frameworku. Framework funguje na báze návrhového modelu MVP (angl. *Model–View–Presenter*) a teda sa v ňom implementuje, ako užívateľské rozhranie, tak aj modely a entity systému. Model zakomponovaný v Nette aplikačnom rámci bližšie priblíži nasledujúca kapitola 5.2.1. Samotné webové stránky Nette dynamicky generuje.

[1] `https://httpd.apache.org/`
[2] `https://doc.nette.org/en/2.x`

### 5.2.1 Nette a model MVP

Nette je objektovo–orientovaný aplikačný rámec, využívajúci softvérový návrhový vzor MVP, *Model–View–Presenter* (podobné modelu MVC - *Model–View–Controller*), zobrazený na obrázku 5.1. Objekty modelu definujú objektové triedy, ktoré sa starajú o logiku aplikácie, ako napríklad komunikáciu s databázou. Pohľady v Nette reprezentujú objekty nazývané šablóny (angl. *templates*). Nette šablóny sú písané v šablónovacom jazyku *Latte* a obsahujú HTML kód, ktorý je možné pomocou špeciálnych značiek, rozšíriť o PHP volania metód a funkcií. Poslednou časťou modelu je *presenter*, ktorý manipuluje s modelom a predáva získané dáta z modelu, šablóne. Presenter tvorí so šablónou hierarchiu presenteru, s ktorou pracuje užívateľ. Rozdielom modelov MVP a MVC spočíva v tom, že pri MVP užívateľská akcia smeruje na hierarchiu presenteru a pri MVC je akcia smerovaná na konkrétny controller [4, 9, 10].




Obr. 5.1: Model MVP.

### 5.2.2 Vykonané zmeny

Pre potreby pripojenia implementovaného modulu, boli do architektúry webového serveru naimplementované tri nové triedy formulárov a ich šablón. Jednalo sa konkrétne o formuláre, ktoré od užívateľa získali vstupné súbory CSV a tým umožnili nahrávanie súborov. Formuláre boli rozšírené o kontrolu, že nahrávaný súbor je len vo formáte CSV, aby sa predišlo chybám. Rovnako tak, tieto formuláre obsahujú akciu pre stiahnutie vzorového CSV súboru, do ktorého môže užívateľ vložiť novú dávku matričných záznamov.

Medzi ďalšie zmeny patrí aj tlačidlo hromadného exportovania aktuálnej tabuľky. Táto funkcionalita prinesie užívateľovi ľahkú možnosť, ako extrahovať celú matriku, ktorú si prezerá.

### 5.2.3 Výstup užívateľovi

Pri importne dát sa môže vyskytnúť chyba vo vstupných dátach. Preto bol implementovaný mechanizmus, ktorý oboznámi užívateľa o úspešnosti spracovania dát. Pri úspešnom spracovaní, systém oznámi užívateľovi počet nových a aktualizovaných matričných záznamov. Pri vyskytnutí chyby, je užívaveľ informovaný okrem počtu vložených dát, aj o počte a miestach chýb, na ktoré sa počas spracovania narazilo.

Informácie sú vyberané z logovacieho súboru, ktorý definuje haš v názve súboru. Haš je počítaná z dátumu a času spustenia spracovania, typu matričného záznamu, identifikátora užívateľa a typu akcie spracovania. K výpočtu tejto haše bol použítý algoritmus MD4 [16]. Algoritmus bol zvolený len na identifikovanie súborov a nie na kryptografické účely.

# Kapitola 6

# Testovanie

Experimentálnou časťou tejto práce je overenie správnosti naimplementovanej funkcionality, ktorú opíše nasledujúca kapitola. Overenie je rozdelené do dvoch podkapitol. Prvá sa venuje procesu overenia kvality spracovania vkladaných dát. Druhá podkapitola opíše spôsob overenia samotnej webovej aplikácie po vykonaných zmenách.

## 6.1 Spracovanie dát

Pre prvú časť testovania boli vytvorené automatizované testy pre moduly spracujúce formáty, použitím vstavaného aplikačného rámca programovacieho jazyka Python, nazývaného „unittest“.

Aplikačný rámec je kolekcia funkcií, modulov a nástrojov, ktoré zjednodušujú vývoj aplikácií. Ponúkajú efektívne riešenia dielčích funkcionalít k tomu, aby vyvíjaná aplikácia splňovala očakávania, ktoré vývojár implementuje. Charakteristickými črtami aplikačného rámca sú vyššia abstrakcia poskytovaná jednotlivými komponentmi, ďalej definícia ako abstrakcie spolu fungujú a znovu použiteľnosť jednotlivých komponentov [5].

Vstavaný testovací aplikačný rámec unittest má formu objektovo–orientovaného nástroja. Ponúka naimplementované metódy, ktorá sa spúšťajú pred a po každom nadefinovanom teste, a ďalej ponúka nadefinovanie testovacích prípadov a testovacích súprav. Ďalej framework implementuje komponentu, ktorá umožňuje naplánovať jednotlivé testy a vypísanie výsledku testov.

Testy je možné spustiť nezávisle a každý test generuje svoje vlastné logovacie správy, popísané v kapitole 4.1.1. Pre oba tieto prípady bol rozšírený súbor *Makefile* o príkazy spustenia všetkých unit testov a odstránenie logovacích súborov, ktoré sa vygenerovali pri testovaní aplikácie.

### 6.1.1 Testovacia trieda

Automatizované testy sú implementované objektovo–orientovaným návrhom a každá testovacia entita má vlastnú triedu.

Trieda obsahuje metódu, ktorá pripraví potrebné časti aplikácie ako napríklad, logovací súbor, objekt mapovania a prázdnu množinu spracovaných dát, prípadne nastaví pripojenie k databáze. Ďalšou elementárnou metódou je metóda, ktorá sa spustí pri skončení každého jedného testu. Táto metóda má na starosti správne ukončenie aplikačných súčastí a vrátenie zmien v testovacej databáze príkazom *rollback*.

Samotné testovacie prípady majú vlastnú metódu v prislúchajúcej triede. Metóda obsahuje prípravu testovacích dát a zaznamenanie aktuálneho stavu aplikácie. Následne nato sa vykoná testovaný proces a vyhodnotenie testu.

## 6.2 Uživateľské rozhranie

Valídnosť pripojenia novej funkcionality na webové rozhranie bolo vykonané ručne. Tento druh testovania bol zvolený z dôvodu, že existujúce webové rozhranie automatické testy neobsahovalo a počet úprav bol tak malý, že vytváranie automatizovaných testov bolo zbytočné. K zadefinovaniu testovacích prípadov, boli navrhnuté testovacie scenáre, ktorú sú súčasťou prílohy D. Scenáre pokrývali testy nahrania a spustenia spracovania vstupného CSV súboru s dátami, ďalej exportu vzorového súboru CSV a súboru s výberom extrahovaných dát.

# Kapitola 7

# Záver

Cieľom tejto práce bolo navrhnúť systém šablón, ktoré by slúžili k popisu logických prepojení databázových štruktúr pri importe a exporte matričných záznamov systému DEMoS. Práca sa zaoberala naštudovaním problematiky a požiadaviek na šablóny. S týmito poznatkami sa vytvorili základy pre implementovanú aplikáciu, následne napojenú na existujúci systém DEMoS.

Výsledkom práce je aplikačný skript, ktorý využíva navrhnutý systém šablón pre transformáciu dát z CSV tabuľky do normalizovanej podoby záznamov relačnej databáze, čím implementuje import dát. Transformácia funguje aj z normalizovanej podoby do súboru s formátom CSV s matričnými dátami, ale aj iba do súboru so záhlavím tabuľky bez dát.

Naimplementovaný modul dokáže pracovať v dvoch verziách. Ako samostatný aplikačný skript, ktorý sa priamo pripája k databáze systému DEMoS a je ovládaný cez prístupový terminál. Druhou verziou je rozšírenie k existujúcemu webovému rozhraniu, ktorý naimplmentový modul spúšťa. Rozšírenie prinesie budúcim užívateľom systému DEMoS možnosť vkladať, upravovať a exportovať uložené matričné záznamy. Oproti súčasnej možnosti manipulácie s dátami po jednom, je rozšírenie veľkým prínosom pre príjemnejšiu prácu s genealogickými informáciami. Súčasťou práce bol vyvinutý pomocný skript, ktorý sa nielen stará o inštaláciu potrebných závislostí ale aj prečistenie vygenerovaných CSV súborov.

K testovaniu všetkých častí boli použité automatizované unit testy a ručné testovanie, definované testovacími scenármi. Automatizované testy sú oddelené od implementovaného kódu, a nemajú vplyv na reálne dáta systému DEMoS, a udržiavanie ich správnosti bude potrebné pri budúcej úprave navrhnutej aplikácie. Testovacie scenáre budú musieť byť upravené pri každej zmene webového užívateľského rozhrania.

Napojením navrhnutého riešenia na systém DEMoS sa vývoj na aplikácii touto prácou nekončí. Navrhnutý modul je možné v budúcnosti prepojiť s automatizovanou analýzou snímok historických matričných záznamov vo vysokom rozlíšení. Priestor na zlepšenie sa nájde aj pri užívateľskom zážitku používania webového rozhrania, napríklad doimplementovaním podrobnejšieho filtrovania matričných záznamov, čím by sa upresnila extrahovaná množina dát, alebo umožnenie vlastného výberu matričných záznamov k exportovaniu.

# Literatúra


[1] Bartůněk, V. Vývoj farních matrik. In: *Časopis rodopisné společnosti v Praze*. 1940, s. 6–17. Dostupné z: http://www.historie.hranet.cz/heraldika/crsc/crsc1940-1.pdf.

[2] Beroušek, F. et al. Ottův slovník naučný: illustrovaná encyklopaedie obecných vědomostí. In:. Československo, Praha: J.Otto, 1900, sv. 16, s. 995. Dostupné z: https://www.digitalniknihovna.cz/nkp/uuid/uuid:2f80dfe0-12af-11e5-9192-001018b5eb5c.

[3] Bray, T. *The JavaScript Object Notation (JSON) Data Interchange Format* [RFC 8259]. RFC Editor, dec 2017. DOI: 10.17487/RFC8259. Dostupné z: https://www.rfc-editor.org/info/rfc8259.

[4] Břešťan, H. *Rozdíly mezi MVC a MVP*. 2014 [cit. 2023-05-03]. Dostupné z: https://devel.cz/otazka/rozdily-mezi-mvc-a-mvp.

[5] Cavaness, C. *Programming Jakarta Struts*. 2. vyd. O'Reilly Media, 2004. ISBN 978-0596006518.

[6] Česká republika. Základy matričného práva. In: *Praktický průvodce a rádce úředníka pro přípravu na matriční zkoušku podle zákona č. 301/2000 Sb., o matrikách, jménu a příjmení a o změně některých souvisejících zákonů, ve znění pozdějších předpisů*. Praha: Ministerstvo vnitra, 2022. ISBN 978-80-7616-129-0. Dostupné z: https://www.mvcr.cz/soubor/zaklady-matricniho-prava-prakticky-pruvodce-a-radce-urednika.aspx.

[7] Collins COBUILD Advanced Learner's Dictionary. *Definition of genealogy* [online]. HarperCollins Publishers, 2023 [cit. 2023-03-27]. Dostupné z: https://www.collinsdictionary.com/dictionary/english/genealogy.

[8] Gellens, R. *The Text/Plain Format Parameter* [RFC 2646]. RFC Editor, august 1999. DOI: 10.17487/RFC2646. Dostupné z: https://www.rfc-editor.org/info/rfc2646.

[9] Grudl, D. *Jak fungují aplikace?* [cit. 2023-04-29]. Dostupné z: https://doc.nette.org/cs/application/how-it-works.

[10] Grudl, D. *Nette Framework: MVC MVP*. 2009 [cit. 2023-05-04]. Dostupné z: https://zdrojak.cz/clanky/nette-framework-mvc-mvp/.

[11] Ingram, D. *Design - Build - Run: Applied Practices and Principles for Production Ready Software Development*. Indianapolis, Ind.: Wrox, 2009. ISBN 9780470257630.

[12] Kočí, R., Rozman, J. a Zbořil, F. 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*. Association for Computing Machinery, 2020, s. 21–25. DOI: 10.1145/3378936.3378974. ISBN 978-1-4503-7690-7. Dostupné z: https://www.fit.vut.cz/research/publication/12114.

[13] Milan. *Matriční zápis sňatku, Wien XIX, Heiligenstadt, matrika z let 1907–1912, fol. 18*. 2015 [cit. 2023-05-01]. Dostupné z: http://jihoceske-rody.eu/rody/wp-content/uploads/2015/02/frolik-petr-hurky-wien-19.jpg.

[14] Minella, M. *Pro Spring Batch*. Apress, 2011. Books for professionals by professionals. ISBN 9781430234531.

[15] Procházková, M. *Matriky a jejich minulost, současnost a budoucnost*. Brno, 2020. Diplomová práce. Masarykova univerzita, Filozofická fakulta. Vedúci práce Smitka, J. Dostupné z: https://is.muni.cz/th/ubws2/.

[16] Rivest, R. L. *The MD4 Message-Digest Algorithm* [RFC 1320]. RFC Editor, 1. apríl 1992. DOI: 10.17487/RFC1320. Dostupné z: https://www.rfc-editor.org/info/rfc1320.

[17] Shafranovich, Y. *Common Format and MIME Type for Comma-Separated Values (CSV) Files* [RFC 4180]. RFC Editor, október 2005. DOI: 10.17487/RFC4180. Dostupné z: https://www.rfc-editor.org/info/rfc4180.

[18] Skřebská, Z. *Genealogie a tvorba rodokmenů*. Brno, 2017. Bakalárska práca. Masarykova univerzita, Přírodovědecká fakulta, Brno. Vedúci práce Malina, J. Dostupné z: https://is.muni.cz/th/mrrs2/.

[19] SOA Třeboň. *Římskokatolický farní úřad Jílovice, matrika narozených 1804-1865*. [cit. 2023-05-01]. Dostupné z: http://rodokmen.nase-koreny.cz/images/ukazky/matriky/1845_nar.jpg.

[20] StackPath, L. *What is a web application?* [online]. [cit. 2023-03-19]. Dostupné z: https://www.stackpath.com/edge-academy/what-is-a-web-application/.

[21] Stephens, R., Plew, R. a Jones, A. D. *Naučte se SQL za 28 dní*. 1. vyd. Computer Press, 2012. ISBN 978-80-251-2700-1.

[22] Van Rossum, G. a Drake, F. L. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN 1441412697.

[23] Welch, A. *Understanding Storage Sizes for MySQL TEXT Data Types*. 2021 [cit. 2023-04-29]. Dostupné z: https://chartio.com/resources/tutorials/understanding-strorage-sizes-for-mysql-text-data-types/.

[24] Wiki, S. *BasicConcepts — SELinux Wiki,*. 2009 [cit. 2023-04-29]. Dostupné z: https://selinuxproject.org/w/?title=BasicConcepts&oldid=805.

[25] Yadav, S. C. a Singh, S. K. *An Introduction to Client Server Computing*. New Age International (P) Ltd., Publishers, 2009. ISBN 978-81-224-2861-2.

# Príloha A

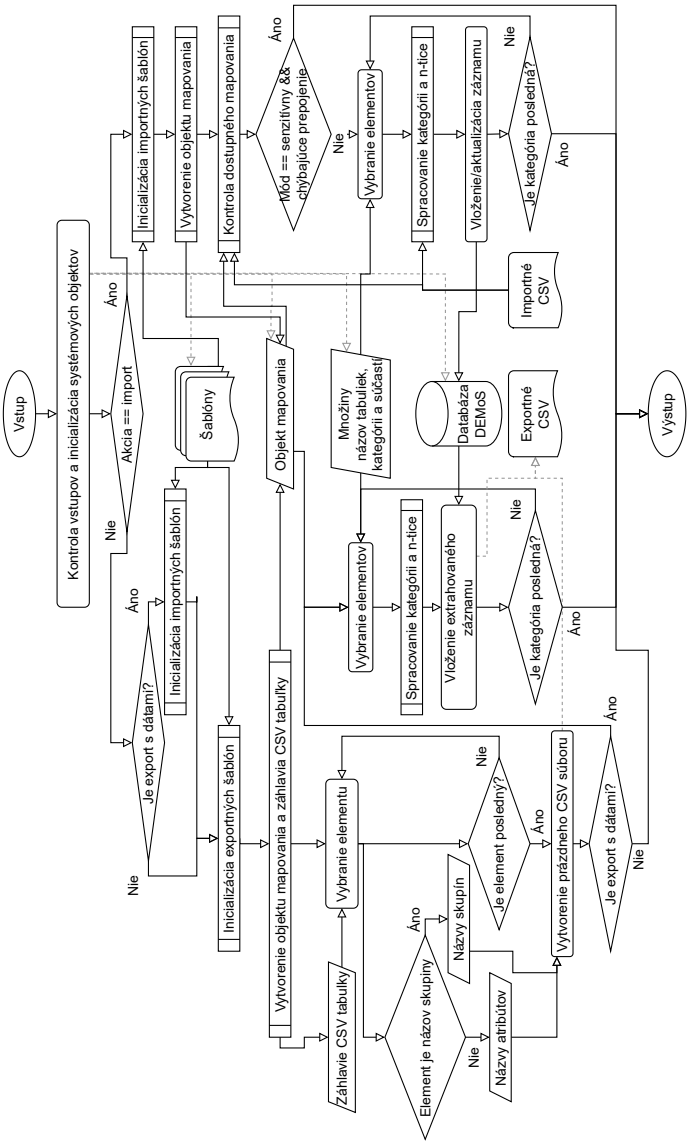
# Obsah priloženého média

Štruktúra vypísaná v tejto prílohe popisuje štruktúru a obsah dôležitých častí priloženého média.

- data
  - demos-db.sql – SQL skript určený k vytvoreniu základnej štruktúry databáze
  - parish-birth.csv – CSV súbor so vzorovými matričnými záznamami narodenia
  - parish-burial.csv – CSV súbor so vzorovými matričnými záznamami úmrtia
  - parish-marriage.csv – CSV súbor so vzorovými matričnými záznamami sobášu
- doc – priečinok obsahujúci dokumentáciu
  - src – priečinok obsahujúci zdrojové súbory pre vytvorenie technickej správy
  - appendix-B.pdf – PDF verzia prílohy B v plnej veľkosti
  - xjanda26-batch-processing-module.pdf – PDF verzia technickej správy
- src – priečinok so zdrojovým kódom modulu
  - definition – priečinok so systémom šablón
    - birth
    - burial
    - marriage
  - config.py – konfiguračný súbor
  - Makefile – súbor obsahujúci popisy a príkazy k inštalácii a čisteniu dát
  - README.md – súbor s inštrukciami k spusteniu a k požiadavkám aplikácie

# Príloha B

# Vývojový diagram modulu




Obr. B.1: Vývojový diagram implementovaného modulu

# Príloha C

# Tabuľky dátových formátov šablóny

| Typ formátu | Skupina formátov | Varianta | |
|---|---|---|---|
| | | Skratka | Popis |
| cudzí kľúč | matričné záznamy | B | Identifikátor záznamu narodenia |
| | | M | Identifikátor záznamu sobášu |
| | | BU | Identifikátor záznamu pohrebu |
| | matričná kniha/listina | R | Identifikátor záznamu matričného dokumentu |
| | užívateľ systému DEMoS | U | Identifikátor záznamu užívateľa |
| | osoba | P | Identifikátor záznamu osoby vo vzťahu k udalosti |
| | vzťah osoby | PR | Identifikátor záznamu normalizovaný typ vzťahu |
| | krstné meno | NM | Identifikátor záznamu mužského krstného mena |
| | | NF | Identifikátor záznamu ženského krstného mena |
| | | NU | Identifikátor záznamu krstného mena bez známeho rodu |
| | | N? | Identifikátor záznamu krstného mena, ktorého rod určuje pohlavie hlavnej osoby záznamu |
| | priezvisko | SM | Identifikátor záznamu mužského priezviska |
| | | SF | Identifikátor záznamu ženského priezviska |
| | | SU | Identifikátor záznamu priezviska bez známeho rodu |
| | | S? | Identifikátor záznamu priezviska, ktorého rod určuje pohlavie hlavnej osoby záznamu |
| | povolanie | O | Identifikátor záznamu normalizovaného názvu povolania |
| | sídlo/bydlisko | DO | Identifikátor záznamu sídla/bydliska |
| | adresa | A | Identifikátor záznamu adresy skladajúci sa z FK bydliska, názvu ulice a čísla orientačného |
| | spôsob úmrtia | BD | Identifikátor záznamu normalizovaného názvu príčiny úmrtia |

Tabuľka C.1: Formáty dát mapovania - časť 1.

| Typ formátu | Skupina formátov | Varianta | |
|---|---|---|---|
| | | **Skratka** | **Popis** |
| číselník (enum) | jazyk | LAN | Určuje jednu z hodnôt množiny jazykov, definovú databázou |
| | pohlavie | SX | Určuje jednu z hodnôt množiny pohlaví, definovú databázou |
| | legitimita | LEG | Určuje jednu z hodnôt množiny legitimity, definovú databázou |
| | typ vzťahu | REL | Určuje jednu z hodnôt množiny vzťahov, definovú databázou |
| | vierovyznanie | REG | Určuje jednu z hodnôt množiny vierovyznaní, definovú databázou |
| malé čisla | číslo | N | Celo číselná hodnota v rozsahu <0;255> |
| | logické hodnoty | B0 | Číselná hodnota z množiny {0;1}, pri neuvedenej hodnote je nastsvená na nulu |
| | | B1 | Číselná hodnota z množiny {0;1}, pri neuvedenej hodnote je nastavená na jedna |
| | | BN | Číselná hodnota z množiny {0;1}, pri neuvedenej hodnote je nastavená na *null* |
| čísla | celé číslo | N | Celo číselná hodnota v rozsahu <-32768;32767> |
| desatinné čísla | desatinné čislo | DE | Číslo s plávajúcou desatinnou čiarkou, určené definícou dĺžky jednotlivých častí v n-tici mapovania |
| znakový formát | znaky | CN | Hodnota znaku alebo krátka postupnosť znakov |
| | pozícia | LAY | Hodnota znaku z množiny {C,L,P} |
| | dátum | D | Postupnosť znakov, ktoré majú formát dátumu DD.MM.YYY |
| | kategória kmotra | LT | Hodnota znaku z množiny {L,T} |
| reťazcové | čas | T | Postupnosť znakov, ktoré majú formát *HH:MM h* alebo *HH h* |
| | reťazec | SN | Množina znakov |

Tabuľka C.2: Formáty dát mapovania - časť 2.

# Príloha D

# Testovacie scenáre

### Testovacia situácia: Validácia formuláru - Nahranie súboru s príponou CSV

**Popis:** Overenie, či formulár správne overuje vstup nahrávania súboru, a akceptuje iba súbory s príponou CSV.
**Testovacie kroky:**

1. Prihláste sa s platnými používateľskými prístupmi.
2. Otvorte matriku s výpisom matričných záznamov.
3. Otvorte formulár cez tlačidlo „Vložení záznamů z CSV“ (názov tlačidla sa môže líšiť v závislosti od typu otvorenej matriky).
4. Vložte do vstupu správny súbor s príponou CSV.
5. Odošlite formulár.
6. Overte, že formulár bol akceptovaný a spracovaný správne.
7. Overte, či je nahraný súbor s príponou CSV rozpoznaný a správne spracovaný.
8. Zopakujte test so súbormi s inými príponami (napr. TXT, XLSX) a overte, či ich formulár odmieta.

### Testovacia situácia: Stav importného tlačidla - Prihlásený používateľ

**Popis:** Overenie, či je tlačidlo formulára aktivované pre prihláseného používateľa.
**Testovacie kroky:**

1. Prihláste sa s platnými používateľskými prístupmi.
2. Otvorte matriku s výpisom matričných záznamov.
3. Skontrolujte, či je tlačidlo „Vložení záznamů z CSV“ aktivované, a či naň možno kliknúť (názov tlačidla sa môže líšiť v závislosti od typu otvorenej matriky).
4. Zopakujte test s rôznymi používateľskými účtami a uistite sa, že tlačidlo zostane aktivované.

### Testovacia situácia: Stav importného tlačidla - Anonymný používateľ

**Popis:** Overenie, či je tlačidlo formulára deaktivované pre anonymného (neprihláseného) používateľa.
**Testovacie kroky:**

1. Otvorte matriku s výpisom matričných záznamov bez prihlásenia.
2. Skontrolujte, či je tlačidlo „Vložení záznamů z CSV“ deaktivované a nedá sa naň kliknúť.
3. Pokúste sa kliknúť na tlačidlo a uistite sa, že nespustí žiadnu akciu.
4. Zopakujte test s rôznymi prehliadačmi a uistite sa, že tlačidlo zostane pre anonymných používateľov deaktivované.

### Testovacia situácia: Chybová hláška formuláru - Prípona súboru

**Popis:** Overenie, či sa pri odoslaní formuláru s neplatnou príponou súboru, zobrazí chybové hlásenie.
**Testovacie kroky:**

1. Prihláste sa s platnými používateľskými prístupmi.
2. Otvorte matriku s výpisom matričných záznamov.
3. Otvorte formulár cez tlačidlo „Vložení záznamů z CSV“ (názov tlačidla sa môže líšiť v závislosti od typu otvorenej matriky).
4. Vložte do vstupu súbor s nesprávnou príponou (napr. TXT, XLSX).
5. Pokúste odoslať formulár.
6. Skontrolujte, či sa zobrazuje chybové hlásenie, že sú povolené iba súbory s príponou CSV.
7. Overte, že nie je spustené žiadne backendové spracovanie a že formulár zostáva nezmenený.

### Testovacia situácia: Modálna obrazovka - Stiahnutie vzorového súboru CSV

**Popis:** Overenie funkčnosti modálnej obrazovky a tlačidla sťahovania súboru CSV.
**Testovacie kroky:**

1. Prihláste sa s platnými používateľskými prístupmi.
2. Otvorte matriku s výpisom matričných záznamov.
3. Kliknite na tlačidlo „Vložení záznamů z CSV“ (názov tlačidla sa môže líšiť v závislosti od typu otvorenej matriky).

4. Skontrolujte, či je zobrazená modálna obrazovka
5. Na modálnej obrazovke nájdite tlačidlo „Stáhnout prázdnej CSV soubor“.
6. Kliknite na tlačidlo „Stáhnout prázdnej CSV soubor“.
7. Overte, či sa súbor CSV stiahne automaticky.
8. Overte, či je stiahnutý súbor v správnom formáte CSV a neobsahuje žiadne záznamy len záhlavie tabuľky.

### Testovacia situácia: Stav exportného tlačidla - Prihlásený používateľ

**Popis:** Overenie, či je tlačidlo exportovania záznamov aktivované pre prihláseného používateľa.
**Testovacie kroky:**

1. Prihláste sa s platnými používateľskými prístupmi.
2. Otvorte matriku s výpisom matričných záznamov.
3. Skontrolujte, či je tlačidlo „Exportovat všechny záznamy tabulky“ aktivované, a či naň možno kliknúť.
4. Zopakujte test s rôznymi používateľskými účtami a uistite sa, že tlačidlo zostane aktivované.

### Testovacia situácia: Stav exportného tlačidla - Anonymný používateľ

**Popis:** Overenie, či je tlačidlo exportovania záznamov deaktivované pre anonymného (neprihláseného) používateľa.
**Testovacie kroky:**

1. Otvorte matriku s výpisom matričných záznamov bez prihlásenia.
2. Skontrolujte, či je tlačidlo „Exportovat všechny záznamy tabulky“ deaktivované a nedá sa naň kliknúť.
3. Pokúste sa kliknúť na tlačidlo a uistite sa, že nespustí žiadnu akciu.
4. Zopakujte test s rôznymi prehliadačmi a uistite sa, že tlačidlo zostane pre anonymných používateľov deaktivované.

### Testovacia situácia: Stiahnutie súboru CSV - Matričné záznamy

**Popis:** Overenie, že funkčnosť tlačidla na automatické stiahnutie súboru CSV s údajmi matričných udalostí.
**Testovacie kroky:**

1. Prihláste sa s platnými používateľskými prístupmi.
2. Otvorte matriku s výpisom matričných záznamov.
3. Kliknite na tlačidlo „Exportovat všechny záznamy tabulky“.
4. Overte, či sa súbor CSV stiahne automaticky.
5. Overte, či stiahnutý súbor obsahuje údaje zo zobrazenej tabuľky v správnom formáte CSV.
6. Zopakujte test s rôznymi typmi matričných záznamov a uistite sa, že stiahnuté súbory CSV zodpovedajú príslušným údajom tabuľky.