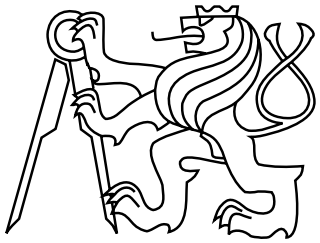
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STAVEBNÍ

BAKALÁŘSKÁ PRÁCE

PRAHA 2009 Pavla ŠMEJKALOVÁ

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STAVEBNÍ
OBOR GEODÉZIE A KARTOGRAFIE



BAKALÁŘSKÁ PRÁCE
VĚCNÁ BŘEMENA V POZEMKOVÝCH ÚPRAVÁCH

Vedoucí práce: Ing. Petr SOUKUP, Ph.D.
Katedra mapování a kartografie

září 2009

Pavla ŠMEJKALOVÁ

ZDE VLOŽIT LIST ZADÁNÍ

Z důvodu správného číslování stránek

ABSTRAKT

Cílem této práce je usnadnit práci zpracovatelům pozemkových úprav. Zpracovatel potřebuje vědět, na kterých parcelách jsou věcná břemena a jména a adresy všech povinných i oprávněných k nim, protože nejen vlastníci pozemků jsou účastníky pozemkových úprav. Tyto informace jsou evidovány v souboru popisných informací katastru nemovitostí a mým úkolem bylo zajistit způsob vyhledávání potřebných informací z něj. Pro vyhledávání dat ze souboru popisných informací jsem zvolila dotazovací jazyk SQL. Výsledné dotazy byly vloženy do programu PROLAND, kde zajišťují výpis změn v osobách povinných i oprávněných a výpis změn na parcelách zatížených věcným břemenem.

KLÍČOVÁ SLOVA

Pozemkové úpravy, věcná břemena, SQL

ABSTRACT

The point of this bachelor work is to make easier job for processers of land adjustment. The processer have to know on which plots are easements and names and addresses of all obliged and entitled persons, because not just owners are participants of land adjustment. These information are filed in the set of description information of land registry and my task was to ensure the way how to find out necessary data from it. For finding out data from the set of decsription information of land registry I chose query language SQL. Final queries were inserted in the program PROLAND, where they ensure the list of changes of obliged and entitled persons and list of changes on plots with easements.

KEYWORDS

Land Adjustment, Easements, SQL

PROHLÁŠENÍ

Prohlašuji, že bakalářskou práci na téma „Věcná břemena v pozemkových úpravách“ jsem vypracovala samostatně. Použitou literaturu a podkladové materiály uvádím v seznamu zdrojů.

V Praze dne ....

(podpis autora)

PODĚKOVÁNÍ

Chtěla bych poděkovat vedoucímu práce Ing. Petru Soukupovi Ph.D. za připomínky a pomoc při zpracování této práce. Dále bych chtěla poděkovat firmě GEPRO spol. s r.o. za poskytnutí testovacích dat a především panu Ing. Michalu Votočkovi Ph.D. za veškerou pomoc a rady.

Obsah

Úvod

Tématem této bakalářské práce jsou věcná břemena v pozemkových úpravách (PÚ) a cílem praktické části je výpis informací o změnách věcných práv a břemen v průběhu pozemkových úprav.

V první kapitole je stručně popsáno co to jsou pozemkové úpravy, podle jakých právních předpisů se provádějí, k čemu jsou dobré a jakým způsobem probíhají. Zároveň je tu řešena problematika věcných břemen jako takových a několik jejich příkladů.

V druhé kapitole je popsána struktura dat z databáze souboru popisných informací katastru nemovitostí (SPI KN).

V následujících kapitolách je popsán dotazovací jazyk SQL, pomocí kterého jsem potřebná data z databáze vybírala a dále je stručně popsán program PROLAND. Pro tento program byly dotazy psány a v něm proběhlo testování k ověření funkčnosti navržených dotazů.

Poslední kapitola se týká přímo realizace zadaného úkolu. Je v ní popsána analýza změn věcných práv a břemen, která ukazuje, k jakým různým změnám věcných práv a břemen může v průběhu pozemkových úprav docházet. Dále je zde podrobně popsán jeden z výsledných SQL dotazů, a sice výpis povinných subjektů k nemovitostem. Nakonec je popsán způsob, jakým byly výsledné dotazy zapracovány do programu PROLAND.

1 Pozemkové úpravy a věcná břemena

1.1 Pozemkové úpravy

Provádění pozemkových úprav se řídí zákonem č. 139/2002 Sb. [1] a vyhláškou 545/2002 Sb. [2].

Pozemkové úpravy obecně slouží k změnám a přesunům pozemků, jejich druhů a způsobu využití, změnám a přesunům vlastnických a jiných práv k nemovitostem. Kvalitně provedené pozemkové úpravy podle [3] přinášejí užitek v mnoha oblastech.

K nim patří:

katastrální úřady – tvorba nových digitálních mapových podkladů a obnova písemného operátu s vyřešenými problémy s vedením parcel ve zjednodušených evidencích

finanční úřady – přehledné podklady pro správu daně z nemovitosti

soudy – jednoznačné definování právních vztahů k nemovitostem

resort životního prostředí a ochrany přírody – přesná evidence kultur, řešení problémů eroze a protipovodňových opatření, zvýšení ekologické stability

zemědělství – podklady pro dotační politiku, optimalizace využití půdního fondu

doprava – evidence skutečného stavu a vlastnictví dopravních staveb i objektů na nich, stavba nových polních cest

Především však pozemkové úpravy přinášejí užitek samotným vlastníkům pozemků. Často totiž lidé nevědí, kde přesně jsou jejich pozemky, ani neznají jejich cenu. Jedním z výsledků pozemkových úprav je to, že z mnoha malých a různě umístěných parcel jednoho vlastníka vznikne obvykle jedna velká vytyčená parcela s příjezdovou cestou. Není-li možné tuto podmínku splnit, vznikne více parcel, ale všechny mají přístupovou cestu a tvar vhodný pro následné využití. Po realizaci pozemkových úprav mohou tak vlastníci své pozemky lépe prodávat, pronajímat a nebo na nich sami hospodařit.

1.1.1 Účastníci pozemkových úprav

Účastníci pozemkových úprav jsou podle [4] tří typů. První jsou orgány státní správy, a to především pozemkový úřad, katastrální úřad, Ministerstvo zemědělství a další dotčené orgány státní správy. Druhou skupinu tvoří vlastníci nemovitostí, osoby s věcnými právy k nemovitostem v obvodu PÚ, obec, správci informačních systémů lesů, komunikací a vodních toků. Třetí skupinou je zpracovatelská geodetická a projekční firma, která je vybrána na základě výběrového řízení.

1.1.2 Formy pozemkových úprav

Formy pozemkových úprav jsou dvojího typu, a to komplexní a jednoduché.

Komplexní pozemkové úpravy jsou vždy se zápisem do katastru nemovitostí (KN), obvodem PÚ je extravilán (část katastrálního území mimo zastavěné území) a řeší se zde i síť polních cest, ochrana půdy, protierozní opatření, ekologická stabilita apod.

Jednoduché pozemkové úpravy se ještě dělí na dva typy. První je bez zápisu do KN, jde jen o rychlé vyčlenění pozemku pro hospodaření. Tento typ byl však po zavedeni zákona 139/2002 Sb. [1] zrušen. Druhý typ je se zápisem do KN a řeší se zde pouze vlastnické vztahy například u nedokončených přídělových a scelovacích řízení a neřeší se přístupové cesty, ochrana půdy apod.

1.1.3 Etapy pozemkových úprav

Průběh pozemkových úprav se dá podle [4] rozdělit na pět základních etap, a sice programovou, přípravnou, projekční, realizační a kontrolní.

Programová etapa

Na starost ji má Pozemkový úřad, který sbírá a vyhodnocuje informace o katastrálních územích a stanovuje naléhavost pozemkových úprav. Dále zjišťuje zájem obcí, vlastníků a nájemců půdy o PÚ. Po vyhodnocení vzniká pořadník katastrálních území. Poté Pozemkový úřad zahájí PÚ, vyhlásí výběrové řízení pro zpracovatele a kontaktuje dotčené orgány státní správy.

Přípravná etapa

Provádí se průzkum území, analýza stávajících podkladů, doplňování bodového pole, určování obvodu PÚ, zjišťování průběhu hranic a vypracování vstupních nároků vlastníků.

Projekční etapa

V této etapě se pojednává plán společných zařízení, projektant zpracovává návrh pozemkové úpravy a dochází k jeho projednání. Pro schválení návrhu je nutný souhlas vlastníků alespoň tří čtvrtin výměry v obvodu PÚ. Poté pozemkový úřad stanoví čas pro připomínky a námitky k návrhu a dochází postupně k prvnímu rozhodnutí – o schválení návrhu PÚ a k druhému rozhodnutí – o výměně a přechodu práv. Proti prvnímu rozhodnutí se lze odvolat. O odvolání rozhoduje Pozemkový úřad nebo Ústřední Pozemkový úřad. Proti druhému rozhodnutí se už odvolat nelze, rozhodnutí se vydává veřejnou vyhláškou, písemně se doručí vlastníkům a na Katastrální úřad.

Realizační etapa

Zpracovatel – geodet má na starosti vytyčení nových hranic pozemků, tvorbu digitální katastrální mapy a nového souboru popisných informací. Pozemkový úřad má na starosti zápis do KN a realizaci společných zařízení. To znamená výstavbu nových polních cest pro zpřístupnění všech pozemků, protierozních opatření pro ochranu půdního fondu, vodohospodářských opatření a územních systémů ekologické stability (biocentra a biokoridory) k ochraně životního prostředí.

Kontrolní etapa

Pozemkový úřad kontroluje čerpání financí ze státního rozpočtu i z jiných zdrojů, kontroluje funkci společných zařízení.

1.1.4 Změny vlastnických a jiných práv k nemovitostem

Vzhledem k obvyklé délce trvání pozemkových úprav (2 až 9 let) dochází v průběhu k mnoha změnám vlastnických a jiných práv k pozemkům v obvodu PÚ. Vlastníci po obdržení vstupních nárokových listů obvykle teprve zjistí, jaké pozemky ve skutečnosti vlastní, zjistí i jejich cenu a mohou tak některé své pozemky například prodat. Tím pádem mezi přípravnou a realizační etapou, tzn. mezi obdržením vstupních nároků a zápisem výsledků pozemkové úpravy do katastru nemovitostí, dochází ke změnám v osobách účastníků PÚ.

Zpracovatel PÚ potřebuje mít přehled o všech účastnících PÚ, protože jeho úkolem je rušit stávající parcely a vytvářet nové a převádět věcná břemena z původních parcel na nově vytvořené, případně již nepotřebná břemena rušit. Proto je nutná dobrá komunikace mezi zpracovatelem pozemkových úprav a pozemkovým a katastrálním úřadem, který veškeré změny eviduje v SPI a aktuální stav SPI zasílá zpracovateli.

1.2 Věcná břemena

Věcné břemeno je legitimní způsob, jak omezovat vlastníka nemovitosti ve prospěch někoho jiného, a to tak, že je povinen něco trpět, něčeho se zdržet, nebo něco konat. Věcné břemeno má obvykle dvě smluvní strany, jedna je povinná (vlastník nemovitosti), druhá je oprávněná (vlastník jiné nemovitosti, osoba fyzická či právnická, v jejíž prospěch je břemeno zřízeno).

Existují dva druhy břemen, a to buď břemena vztažená na nemovitost nebo na osobu. Při převodu vlastnických práv u nemovitosti zatížené břemenem, přechází na nového vlastníka i povinnosti týkající se daného břemene. V případě věcného břemene vtaženého k určité osobě, ať už právnické či fyzické nemohou práva přejít na novou osobu.

Podle zákona č. 40/1964 Sb. [5] patří věcná břemena mezi věcná práva. Věcná práva jsou práva absolutní, což znamená že působí proti všem. Tvoří skupinu občanskoprávních vztahů a představují úplné nebo částečné panství nad konkrétní věcí. Mezi věcná práva patří vlastnické právo, držba a věcná práva k cizí věci (jiná věcná

práva), což jsou věcná břemena, zástavní právo, podzástavní právo a zadržovací právo.

Typy jiných věcných práv mohou být například: věcné břemeno cesty, chůze, jízdy, vedení, spoluužívání, braní vody, výměnku, užívání; zástavní práva ve prospěch bank a spořitelen; nařízení k exekuci; předkupní práva; omezení dispozičních práv atd.

1.2.1 Příklady věcných břemen

Příklad pro věcné břemeno vztažené k parcele může být třeba věcné břemeno cesty. K parcele A není přístupová cesta, je tedy nutné chodit přes parcelu B. V takovémto případě je vhodné sepsat smlouvu o věcném břemeni chůze, kde oprávněná je parcela A a povinná je parcela B. Při změně vlastníků kterékoli z obou parcel břemeno zůstává v platnosti.

Příklad pro věcné břemeno vztažené na osobu je třeba věcné břemeno výměnku. Paní C prodá svůj vlastnický podíl svému vnukovi - panu D, ale přesto chce dále v domě bydlet. Je vhodné v takovém případě uzavřít smlouvu o věcném břemeni výměnku. Paní C (oprávněný) tak má jistotu, že i kdyby pan D (povinný) dům prodal, ona tam stále může žít a nový vlastník ji nemůže vystěhovat. Toto břemeno trvá tak dlouho, dokud se osoba A neodstěhuje a nezruší břemeno a nebo dokud nezemře. Znamená to tedy, že povinná strana se prodejem nemovitosti změnit může, oprávněná však ne.

Další příklad věcného břemene vztaženého na osobu může být zástavní právo. Vlastník parcely E má u banky F půjčku, za jejíž splacení ručí bance F parcelou E. Pro banku tak vzniká zástavní právo k parcele E. Parcela E je tak povinná strana, banka F je oprávněná.

2 Struktura SPI

Databáze katastru nemovitostí obsahuje mnoho navzájem propojitelných tabulek. Pro výběr potřebných dat pro zjištění parcel s věcným břemenem a osob vlastníků, povinných a oprávněných jsou nutné tyto tabulky: `PARCELY`, `KATUZE`, `JINE_PRAV_VZTAHY`, `TELESA`, `BUDOVY`, `VLASTNICTVI`, `JEDNOTKY` a `OPRAV_SUBJEKT`.

Na obrázku č. B.1 v příloze jsou zobrazené ty tabulky, které přímo vstupují do výběrového dotazu a jsou na něm znázorněny vzájemné vztahy mezi nimi.

2.1 Tabulka OPRAV_SUBJEKT

Tabulka Oprávněných subjektů popisuje všechny osoby s právy k nemovitostem. Podle sloupce `OPSUB_TYPE` se odlišuje zda se jedná o osobu právnickou, fyzickou a nebo o společné jmění manželů (SJM). U právnických osob zde je uložen jejich název, IČO a adresa sídla, u fyzických osob zde je jméno, příjmení, tituly před i za jménem, rodné číslo a adresa trvalého bydliště.

Způsob uložení SJM

Problematické je vyhledávání osob s právy k parcelám, budovám či jednotkám jedná-li se o společné jmění manželů. Podle dřívějších právních předpisů se to nazývalo bezpodílové spoluvlastnictví manželů (BSM) a pod touto zkratkou je stále na katastru evidováno. Každý oprávněný subjekt, tedy oba manželé, jsou evidování pod jiným identifikačním číslem v tabulce `OPRAV_SUBJEKT` a pod dalším identifikačním číslem jsou evidováni jako BSM, kde je pak ve sloupci `ID_JE_1_PARTNER_BSM` identifikační číslo jednoho partnera a ve sloupci `ID_JE_2_PARTNER_BSM` je identifikační číslo druhého partnera.

2.2 Tabulka JINE_PRAV_VZTAHY

Tabulka Jiných právních vztahů popisuje koho, nebo čeho se týkají věcná břemena, nebo jiná věcná práva. Na jednom řádku je zde identifikační číslo oprávněné i povinné

strany, popis práva a typ právního vztahu. Rozlišit o jakou stranu, zda povinnou či oprávněnou, tak lze pouze podle sloupce. Do sloupců obsahující `_K` se zapisují identifikační čísla povinné strany, do sloupců `_PRO` se zapisují identifikační čísla oprávněné strany. Zjednodušená ukázka uložení dat v tabulce Jiných právních vztahů je zobrazena v tabulce 2.1 na stejných příkladech, jaké byly použity v kapitole 1.2.1 o věcných břemenech.

PAR_ID_K	PAR_ID_PRO	OPSUB_ID_K	OPSUB_ID_PRO	POPIS
B	A			chůze
D			C	výměnek
E			F	zástavní právo

Tab. 2.1: Ukázka tabulky JINE_PRAV_VZTAHY

3 Jazyk SQL

Jazyk SQL je podrobně popsán v řadě prací (např. [6]), proto se o něm zmíním jen okrajově.

Historie tohoto jazyka sahá do sedmdesátých let, kdy se ve firmě IBM zabývali problematikou relačních databází. Vytvořili proto sadu příkazů pro ovládání relačních databází, a sice jazyk SEQUEL (Structured English Query Language). Podle názvu lze odhalit, že se tvůrci snažili syntaxi tohoto jazyka co nejvíce přiblížit běžnému jazyku – angličtině. Postupem času se název zkrátil na SQL a tento jazyk se stal standardem pro práci s databázemi. Ačkoli různé firmy vyvíjeli různé systémy řízení databází, většinou se syntaxe dotazů v jednotlivých programech liší jen minimálně.

Aplikace PROLAND (více v kapitole 4), pro kterou jsem dotazy formulovala, používá syntaxi MS ACCESS.

Jazyk SQL byl vytvořen přímo pro správu a údržbu databází a není proto vhodný pro obecné programování aplikací. Dá se však s výhodou použít v kombinaci s jiným programovacím jazykem (Basic, C++, Java), kde zajišťuje manipulaci s datovými strukturami.

3.1 Kategorie příkazů jazyka SQL

Jednotlivé příkazy jazyka SQL se mohou podle své funkce rozdělit do několika kategorií:

- jazyk DDL (Data Definition Language)
- jazyk DQL (Data Query Language)
- jazyk DML (Data Manipulation Language)
- jazyk DCL (Data Control Language)

3.1.1 Jazyk DDL

Do kategorie DDL patří příkazy, umožňující tvorbu a úpravy datových objektů, například tabulky nebo indexy. Klíčová slova této kategorie jsou CREATE, ALTER a DROP.

CREATE

Tento příkaz umožňuje vytvářet nový databázový objekt. Je třeba definovat jakého datového typu nový objekt bude. Základní datové typy jsou: textový řetězec, číslo, datum a booleovský operátor. Datových typů existuje mnoho a různé systémy řízení báze dat mají různé definice pro stejné typy.

ALTER

Klíčové slovo ALTER umožňuje změnit datový objekt. Použitím tohoto slova je možno přidávat sloupce, měnit datové typy již navržených sloupců, přejmenovat sloupce apod.

DROP

Pomocí klíčového slova DROP se ruší datový objekt toho typu, jaký je v příkazu uveden.

3.1.2 Jazyk DQL

Jedná se o nejčastěji užívanou kategorii jazyka SQL. Příkazy z této kategorie slouží k načítání dat z databáze. Základní klíčové slovo je SELECT, které se vyskytuje ve všech příkazech kategorie DQL. Dále se v příkazech používají především slova FROM, WHERE, ORDER BY.

SELECT

Uvádí sloupce, které chce uživatel zobrazit. Názvy jednotlivých sloupců se oddělují čárkami. Chceme-li vypsat všechny sloupce tabulky, stačí napsat symbol * za klíčové slovo SELECT. Pokud se ve vybraných sloupcích vyskytují duplicitní řádky a

nechceme je zobrazit, použije se za klíčové slovo SELECT slovo DISTINCT, které duplicitní řádky vyloučí.

FROM

Uvádí název tabulky ze které jsou data vyhledávána. Chceme-li do výsledku vypsat data z více tabulek, je nutno tabulky vzájemně propojit. K tomu slouží slovo JOIN. Podle způsobů spojení tabulek se odlišují možnosti INNER JOIN, LEFT JOIN, RIGHT JOIN, CROSS JOIN. Je zde ale nutné vypsat pomocí kterých sloupců se tabulky propojí. Jako příklad uvedu dotaz, jehož výsledkem je seznam čísel listů vlastnictví a k němu odpovídající výměry jednotlivých parcel:

```
SELECT P.VYMERA_PARCELY, T.CISLO_TEL
FROM PARCELY P INNER JOIN TELESA T ON P.TEL_ID = T.ID
ORDER BY T.CISLO_TEL
```

WHERE

Zatímco klíčové slovo SELECT určí jaké sloupce se zobrazí, slovo WHERE vybere požadované řádky. Co se zobrazí, se určí podle Booleovské logiky. Ve výsledku jsou zobrazeny ty řádky, pro které je vyhodnocení klauzule WHERE logická hodnota „true“. Kromě operátoru „rovná se“ se dají použít i jiné logické operátory, jako například „nerovná se“, „menší“, „větší“... Pro formulaci složitějších podmínek výběru se dají použít i spojovací operátory AND nebo OR. Operátor AND znamená, že pro všechny podmínky musí platit výsledek „true“. Operátor OR znamená, že výsledek „true“ musí platit alespoň pro jednu podmínku.

ORDER BY

Umožňuje seřadit výslednou tabulku podle vybraného jednoho nebo více sloupců.

3.1.3 Jazyk DML

Příkazy z kategorie DML umožňují přidávat, měnit nebo odebírat data z databáze. Klíčová slova jsou INSERT, UPDATE, DELETE.

INSERT

Pomocí klíčového slova INSERT lze přidávat řádky do tabulky. Co se do nového řádku uloží je možné napsat výčtem a nebo vnořením příkazu SELECT.

UPDATE

Umožňuje aktualizovat data v databázi. V aktualizačním příkazu je kromě slova UPDATE třeba použít klíčové slovo SET, za které se zadá nová hodnota.

DELETE

Pomocí klíčového slova DELETE se odebírají jeden nebo více řádků tabulky. Při použití tohoto slova, není možné odstranit data jen z některých sloupců, řádek je vždy smazán celý.

3.1.4 Jazyk DCL

V této kategorii se vyskytují příkazy, které dovolují správcům řídit přístup k datům v databázi. K tomu se používají klíčová slova GRANT a REVOKE.

GRANT

Pomocí klíčového slova GRANT uděluje správce databáze jejím uživatelům oprávnění k práci s daty.

REVOKE

Klíčové slovo REVOKE je opakem předcházejícího slova, znamená to, že umožňuje přístupová práva k datům odebírat.

3.2 Syntaktická konvence jazyka SQL

Každý příkaz začíná klíčovým slovem, kterým obvykle bývá nějaké dějové sloveso v anglickém jazyce. Základním a nejpoužívanějším je sloveso SELECT, což znamená „vyber“. Struktura zápisu se podobá anglické větě.

Jednotlivé příkazy se zapisují „volným“ způsobem, tedy neexistují závazná pravidla pro umístění jednotlivých prvků, či pro rozdělování příkazů na řádky, ale je vhodné psát dotaz co nejpřehledněji. Obvykle není nutné psát dotazy velkými písmeny, protože většina programů před zpracováním všechny znaky na velké převede. Je jen málo systémů řízení báze dat které dokáží odlišovat velká a malá písmena.

4 PROLAND

PROLAND je nadstavba programu KOKEŠ vyvíjená firmou GEPRO spol. s r.o. určená pro zpracování a projektování pozemkových úprav a pracovníkům pozemkového úřadu.

Tento program obsahuje sadu funkcí vytvořených přímo pro automatizované zpracování pozemkových úprav. Navíc plně podporuje nový výměnný formát VFK katastru nemovitostí jak pro vstup podkladů, tak i pro export výsledků.

4.1 Vstupní poklady

Při tvorbě návrhu PÚ je třeba jako vstupní grafické podklady vložit do programu vektorovou katastrální mapu - mapa KN, mapu s parcelami pocházejícími z dřívějších evidencí např. evidence nemovitostí, pozemkový katastr nebo z jiných podkladů jako jsou grafické příděly - mapa PK, mapu obvodu PÚ, mapu BPEJ (bonitovaných půdně ekologických jednotek), mapu porostů, mapu kostry (plán společných zařízení). Označení PK zde neznamená pouze parcely pocházející z pozemkového katastru, ale je to označení pro všechny parcely evidované zjednodušeným způsobem. Nejsou-li tyto mapy k dispozici ve vektorové podobě, lze je přímo v programu zvektorizovat.

Vytvořené vektorové mapy musí být topologicky správné, v programu PROLAND jsou funkce kontrolující topologickou čistotu.

Kromě mapových podkladů program potřebuje mít k dispozici ještě databázi souboru popisných informací katastru nemovitostí ve formátu MDB. Tato databáze vznikne importem VFK do programu.

4.2 Tvorba vstupních nároků

K automatizovanému výpočtu vstupních nároků slouží funkce „Průnik map“. Dojde zde k vytvoření grafického průniku mapových podkladů a k naplnění pracovní databáze. Ke každé ploše vzniklé průnikem existuje záznam v pracovní databázi

s informacemi, ke které parcele KN a ke které parcele zjednodušené evidence plocha patří a zda je v obvodu PÚ nebo mimo obvod. Dále zkopíruje potřebné údaje z databáze SPI KN.

Při opakovaném vyvolání funkce „Průnik map“ se zobrazí přehled změn údajů SPI parcel dotčených katastrálních území a přehled změn vlastníků parcel dotčených pozemkovou úpravou. Ve výpisu jsou zobrazeny parcely, u kterých došlo k nějaké změně, tzn. byly zrušeny nebo nově přibyly. Tyto změny upozorní zpracovatele PÚ na proběhlé převody nemovitostí, opravy chyb, nové geometrické plány apod. Přehled změn vlastníků zobrazí osoby, které pozbyly vlastnické právo k nemovitosti, nově vlastnické právo nabyly nebo u nich došlo k nějaké změně (jména, adresy, vlastnického podílu...).

Přehled všech těchto změn lze exportovat do formátu XLS, MDB, HTML nebo TXT a tak je možnost snadno se vrátit ke změnám z dřívějších časových období.

Tabulka kontrol

Funkce „Kontrola“, která obvykle následuje po funkci „Průnik map“, slouží k porovnávání výskytu a výměr parcel v mapě a v databázi. K těmto rozdílům může dojít například při vektorizaci mapového podkladu při špatné čitelnosti parcelních čísel v rastru. Další rozdíly mohou vzniknout, je-li na některé parcele změna zaznamenaná formou geometrického plánu. Program v tomto případě umožňuje opravit údaje SPI u dané parcely, případně ji odstranit, byla-li geometrickým plánem zrušena.

Vzdálenosti parcel

Jako součást vstupních nároků je také třeba vypočítat průměrnou vzdálenost parcel. Program tuto vzdálenost počítá jako vzdušnou vzdálenost těžiště parcely a zvoleného referenčního bodu pro daný list vlastnictví. Pokud není referenční bod zadán, použije se k výpočtu referenční bod katastrálního území (například věž kostela).

Nastavení cen

Funkce „nastavení cen“ umožňuje vybrat, zda se konkrétní parcely budou oceňovat podle kódů z mapy BPEJ v kombinaci s mapou porostu a způsobu využití a nebo

podle ceny za m^2.

Dále se v této funkci dají nastavit hodnoty opravných koeficientů. První zohledňuje rozdíl mezi výměrou vypočtenou ze souřadnic podle zaměření skutečného stavu a součtem výměr podle SPI. Pomocí druhého koeficientu se zmenšují vstupní nároky o výměru nutnou pro společná zařízení. První koeficient lze v programu automaticky vypočítat.

4.3 Seznamy parcel, LV a vlastníků

Program obsahuje sadu funkcí, které umožní vytvářet seznamy:

- abecední rejstřík vlastníků dotčených pozemkovou úpravou,
- abecední rejstřík vlastníků dotčených rozdělením spoluvlastnictví,
- abecední rejstřík jiných oprávněných dotčených pozemkovou úpravou (osoby, které mají jiné věcné právo k parcele dotčené pozemkovou úpravou nebo k budově na takové parcele),
- abecední rejstřík jiných povinných dotčených pozemkovou úpravou (osoby, které vlastní nemovitost zatíženou věcným břemenem ve prospěch nějaké parcely dotčené pozemkovou úpravou nebo ve prospěch budovy na takové parcele),
- seznam listů vlastnictví dotčených pozemkovou úpravou,
- seznam listů vlastnictví dotčených rozdělením spoluvlastnictví,
- seznam parcel dotčených pozemkovou úpravou,
- seznam parcel mimo obvod pozemkové úpravy,
- seznam nově navržených parcel,
- seznam řešených parcel s věcným břemenem,
- seznam řešených parcel s jiným věcným právem,
- přehled bonit vlastnických parcel zemědělské půdy v obvodu pozemkové úpravy,

- přehled bonit nově navržených parcel zemědělské půdy.

Vytvořené seznamy jsou hypertextové, lze tedy „kliknutím“ na parcelní číslo, číslo budovy, nebo číslo LV získat podrobné informace o parcelách, budovách a nebo získat výpis listu vlastnictví. Dále je tu možné vygenerovat nárokové listy (soupis nových pozemků, srovnávací tabulky) pro všechny vlastníky.

Vytvořené seznamy je možné uložit do formátu RTF, XLS, nebo MDB.

4.4 Rebonitace

Rebonitace je souhrn činností spojených s aktualizací mapy BPEJ. Program PROLAND lze použít pro automatické přiřazení kódů BPEJ jednotlivým parcelám a pro výpočet výměr bonitních dílů parcel. Výsledek je možno exportovat do formátu VFK a zaslat katastrálnímu úřadu pro aktualizaci SPI.

Postup přiřazování kódů BPEJ je obdobný jako tvorba vstupních nároků, pouze se po funkci „Průnik map“ spustí funkce „Rebonitace“. Dají se pak zobrazit parcely, u kterých došlo ke změně BPEJ a jako změnové věty lze výsledek exportovat ve formátu VFK.

4.5 Projektování

Program nabízí editační funkce výkresu, které umožňují navrhovat nové parcely, editovat je a nebo je rušit. V průběhu navrhování nové parcelace se zobrazuje pro všechny listy vlastnictví celkový nárok a procenta aktuálního vypořádání návrhu. Průběžně se nároky přepočítávají, lze tak najít nejvhodnější podobu výsledných parcel.

Je-li návrh hotový, přistoupí se ke kontrolám výkresu. Tyto kontroly odstraní případné rozdíly mezi pracovní databází a výkresem a ověří, zda návrh bude možné převést na digitální katastrální mapu.

Kontroluje se zde uzavřenost obvodu parcel, přiřazení parcel na LV, umístění a jedinečnost nově navržených parcelních čísel a provedou se automatické opravy zajišťující topologickou čistotu výkresu. Po těchto opravách se znovu vypočítají výměry,

ceny a vzdálenosti parcel.

4.6 Export VFK

Program umožňuje výsledek pozemkové úpravy exportovat ve formátu změnových vět VFK.

Soubor změnových vět obsahuje: seznam rušených parcel, změny výměry, způsobu určení výměry, druhu pozemku, způsobu využití pozemku, mapového listu, listu vlastnictví, seznam způsobů ochrany nemovitosti, seznam kódů BPEJ a výměr bonitních dílů a seznam nových parcel.

5 Realizace

V této kapitole je popsána stěžejní část mé bakalářské práce.

5.1 Analýza změn věcných práv a břemen

Nejprve bylo třeba zjistit k jakým změnám vlastně vůbec může dojít.
Jsou to:

- vznik nového břemene (například nová hypotéka s ručením nemovitostí),
- zánik stávajícího (například splacením celé dlužné částky u hypotéky),
- změna jména vlastníka, povinného, oprávněného (změna jména po svatbě, změna názvu právnické osoby, prodej nemovitosti jiné osobě),
- změna adresy vlastníka, povinného, oprávněného (změna trvalého bydliště, prodej nemovitosti jiné osobě),
- změna vlastnického podílu k nemovitosti (dokoupení části podílu, dědické řízení),
- změna popisu břemene (například u věcného břemene zástavy pro úvěr se může splacením části dlužné částky snížit ručící částka),

Kromě těchto změn vázaných na osobu vlastníka, oprávněného nebo povinného může dojít i ke změně parcelního čísla parcely s věcným břemenem. To se může stát rozdělením nebo scelením parcely geometrickým plánem.

Dále bylo třeba zjistit co všechno je třeba v databázi vyhledat a určit nejvhodnější podobu výstupu. Pro zjištění této informace jsem byla konzultovat s projektanty pozemkových úprav z firmy Gepard. Podle jejich připomínek jsem jako nejlepší a nejnázornější vybrala dva výstupy. Jedna výstupní tabulka je řazená podle parcel s věcným břemenem, druhá podle vlastníků, jiných povinných a jiných oprávněných.

První tabulka podchytí změny týkající se parcel, druhá změny týkající se osob.

5.2 Popis dotazů

Všechny výsledné SQL dotazy jsou uvedeny v příloze A, zde pro ukázku podrobně popíšu SQL dotaz na výpis povinných osob k parcelám, budovám a jednotkám v obvodu PÚ.

Výběr všech parcel katastru nemovitostí z obvodu PÚ

```
SELECT pkn.katuze, pkn.parskup, pkn.parcis, pkn.parpod, pkn.pardil,
pkn.cislo_lv, pkn.par_id, max(cp.zajem_upr) as ZU
INTO [E:\zalany\pokus\pokus1.mdb].tmp_par
FROM (parcely_kn pkn inner join KU on (KU.katuze = pkn.katuze))
inner join casti_parcel cp on (pkn.id = cp.parcela_kn)
WHERE EXISTS (SELECT 'i' FROM casti_parcel cp
WHERE cp.parcela_kn = pkn.id AND cp.zajem_upr > 0)
group by pkn.katuze, pkn.parskup, pkn.parcis, pkn.parpod, pkn.pardil,
pkn.cislo_lv, pkn.par_id
```

Tento dotaz vybere číslo katastrálního území, způsob číslování parcel, parcelní číslo, podlomení parcelního čísla, díl parcely, číslo listu vlastnictví a identifikační číslo všech parcel katastru nemovitostí v obvodu PÚ a uloží je do pomocné tabulky `tmp_par`.

Výběr všech parcel pozemkového katastru pod nevlastnickými parcelami v obvodu PÚ

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_par
SELECT ppk.katuze, ppk.parskup, ppk.parcis, ppk.parpod, ppk.pardil,
ppk.cislo_lv, ppk.par_id, max(cp.zajem_upr) as ZU
FROM (parcely_pk ppk inner join KU on (KU.katuze = ppk.katuze) )
inner join casti_parcel cp on (ppk.id = cp.parcela_pk)
WHERE EXISTS (SELECT 'i' FROM casti_parcel cp INNER JOIN parcely_kn k
ON (cp.parcela_kn = k.id)
WHERE cp.parcela_pk = ppk.id AND cp.zajem_upr > 0
```

```
AND (k.cislo_lv < 1 OR k.cislo_lv IS NULL))
group by ppk.katuze, ppk.parskup, ppk.parcis, ppk.parpod, ppk.pardil,
ppk.cislo_lv, ppk.par_id
```

Výsledek tohoto dotazu se uloží do stejné tabulky jako předchozí a utvoří tak jedinou tabulku se všemi parcelami v obvodu PÚ. Zda se jedná o parcelu katastru nemovitostí nebo o parcelu pozemkového katastru se pozná podle sloupce parskup (1 odpovídá stavební parcele KN, 2 odpovídá pozemkové parcele KN, 4 stavební parcele PK a 5 pozemkové parcele PK).

Seznam budov na parcelách KN

```
SELECT DISTINCT k.bud_id, t.katuze
INTO [E:\zalany\pokus\pokus1.mdb].tmp_bud
FROM parcely_kn k
WHERE EXISTS (SELECT 'i' FROM [E:\zalany\pokus\pokus1.mdb].tmp_par t
WHERE (t.par_id = k.par_id)) AND k.bud_id IS NOT NULL
```

Tato část vybere číslo katastrálního území a identifikační čísla všech budov, které leží na parcelách KN, která jsou dotčená pozemkovou úpravou a zapíše je do pomocné tabulky `tmp_bud`.

Jiní povinní vůči parcelám

Následující dotazy zapisují do pomocné tabulky `tmp_jpv` identifikační číslo povinné osoby, číslo katastrálního území, označení typu, identifikační číslo oprávněné parcely a typ právního vztahu

```
SELECT DISTINCT j.opsub_id_k AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
FROM jine_prav_vztahy j
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_k = t.par_id)
WHERE j.opsub_id_pro IS NOT NULL
```

Tímto dotazem se do tabulky tmp_jpv zapisují data v případě, že se v databázi SPI vyskytuje identifikační číslo oprávněné osoby.

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT j.opsub_id_k AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM jine_prav_vztahy j
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro = t.par_id)
WHERE j.opsub_id_k IS NOT NULL
```

Tento dotaz zajistí výpis dat pro pro povinné osoby.

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_kn k
on (k.cislo_lv = v.cislo_lv) AND (k.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_k = k.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro = t.par_id)
```

Tento dotaz připíše do tabulky tmp_jpv data, když bude identifikační číslo povinné parcely v tabulce parcely_kn a zároveň identifikační číslo oprávněné parcely bude v pomocné tabulce tmp_par.

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_pk p
on (p.cislo_lv = v.cislo_lv) AND (p.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_k = p.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro = t.par_id)
```

Tímto dotazem se připíší data, jedná-li se o povinnou parcelu z tabulky `parcely_pk`.

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join budovy b
on (b.cislo_tel = v.cislo_lv) AND (b.katuze_kod = v.katuze))
inner join jine_prav_vztahy j on (j.bud_id_k = b.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro = t.par_id)
```

Tento dotaz připíše data, je-li jako povinná strana vedena budova.

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join jednotky jed
on (jed.cislo_lv = v.cislo_lv) AND (jed.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.jed_id_k = jed.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro = t.par_id)
```

A jako poslední varianta je zde výpis dat, kdy je jako povinná strana vedena jednotka.

Jiní povinní vůči budovám

Následující dotazy jsou obdobné předcházejícím, pouze je jako oprávněná strana vedena budova. Výsledek se ukládá do stejné tabulky `tmp_jpv`, rozdíl mezi těmito dvěma typy se pozná podle sloupce `TYP`.

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT j.opsub_id_k AS ID, t.katuze, 32 AS TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
```

```
FROM jine_prav_vztahy j
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id) WHERE j.opsub_id_k IS NOT NULL

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 32 as TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_kn k
on (k.cislo_lv = v.cislo_lv) AND (k.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_k = k.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id)

{INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 32 as TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_pk p
on (p.cislo_lv = v.cislo_lv) AND (p.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_k = p.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 32 as TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join budovy b on (b.cislo_tel = v.cislo_lv)
AND (b.katuze_kod = v.katuze))
inner join jine_prav_vztahy j on (j.bud_id_k = b.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id)
```

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 32 as TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join jednotky jed
on (jed.cislo_lv = v.cislo_lv) AND (jed.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.jed_id_k = jed.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp\bud t
on (j.bud_id_pro = t.bud_id)
```

Výpis jednotlivých osob

Tato trojice dotazů propojí předchozí pomocnou tabulku `tmp_jpv` s tabulkou oprávněných subjektů a tím umožní vypisovat data o jednotlivých osobách. Výběr se uloží do pomocné tabulky `tmp_op`. V druhém a třetím dotazu dojde k rozdělení SJM na jednotlivé osoby.

```
SELECT j.id, j.katuze, j.typ, j.typrav_kod, j.obj_id
INTO [E:\zalany\pokus\pokus1.mdb].tmp_op
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.id = o.id)
WHERE o.opsub_type <> 'BSM'
```

První vypíše všechny osoby které nemají společné jmění manželů.

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_op
SELECT o.id_je_1_partner_bsm, j.katuze, j.typ, j.typrav_kod, j.obj_id
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.ID = o.id)
WHERE o.opsub_type = 'BSM'
```

Druhý vypíše osoby, které jsou vedeny jako první partner společného jmění manželů.

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_op
SELECT o.id_je_2_partner_bsm, j.katuze, j.typ, j.typrav_kod, j.obj_id
```

```
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.ID = o.id)
WHERE o.opsub_type = 'BSM'
```

Třetí vypíše osoby, které jsou vedeny jako druhý partner SJM.

Závěrečný výpis

```
SELECT DISTINCT o.id, t.katuze, o.opsub_type, o.ico, o.doplnek_ico,
o.nazev, o.rodne_cislo, o.titul_pred_jmenem, o.jmeno, o.prijmeni,
o.titul_za_jmenem, o.cislo_domovni, o.cislo_orientacni,
o.nazev_ulice, o.cast_obce, o.mestska_cast, o.obec, o.okres,
o.stat, o.psc, t.typ
FROM [E:\zalany\pokus\pokus1.mdb].tmp_op t
inner join oprav_subjekty o ON (t.id = o.id)
ORDER BY o.id
```

Závěrečný dotaz vypíše informace o všech osobách z pomocné tabulky `tmp_op`. Vypíše se zde zda se jedná o fyzickou nebo právnickou osobu. U právnických osob se zobrazí název, adresa a IČO, u fyzických osob jméno, příjmení, tituly před i za jménem, rodné číslo a adresa trvalého bydliště.

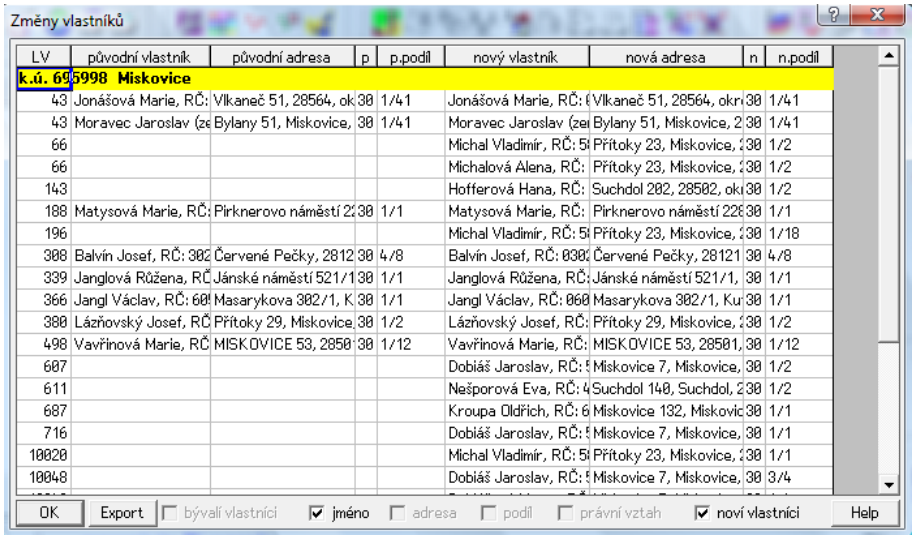
5.3 Rozšíření programu PROLAND

Do programu jako data SPI vstupuje přímo databáze SPI KN, která je však upravena pro snazší práci s ní. Některé tabulky jsou přímo kopírovány, jako například tabulky oprávněných subjektů, jiných právních vztahů a podobně, některé jsou upraveny. Především je upravena tabulka parcely, která je rozdělena na dvě, a to `parcely_kn` a `parcely_pk`. Toto rozdělení je z důvodů přehlednosti a jednodušší práce. V SPI KN jsou všechny parcely vedeny v jediné tabulce, ale vzhledem k tomu, že téměř všude se mapy katastru nemovitostí a pozemkového katastru překrývají, je pro práci s nimi lepší je rozdělit. Dále je přejmenovaná tabulka `TELESA` na `LV`.

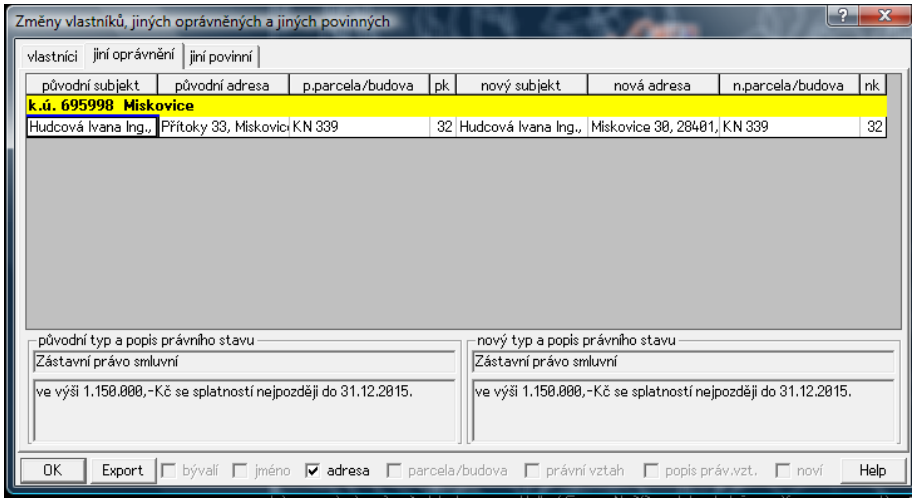
Vložením nové databáze z dalšího časového období (zpravidla další měsíc) se vytvoří v programu tabulky `_zal` s daty předchozího období. Dotazy SQL byly tedy napsány dvakrát, jednou pro nová data, podruhé pro data `_zal` a díky tomu pak lze porovnávat změny v jednotlivých časových úsecích a zapracovávat je do návrhu pozemkových úprav.

Rozšíření aplikace PROLAND proběhlo zapojením mých SQL dotazů do skriptu programu.

Funkce pro zobrazení změny jména a adresy vlastníka a změny vlastnického podílu v programu PROLAND již byla. Pro zobrazení změn v osobách jiných povinných a jiných oprávněných se použila stejná tabulka, pouze byla rozšířena pomocí záložek. Na obrázku 5.1 je ukázka výstupní tabulky změn vlastníků z předchozí verze, na obrázku 5.2 je ukázka nové, rozšířené tabulky.



Obr. 5.1: Vlastníci



Obr. 5.2: Jiní oprávnění

Závěr

Cílem této práce bylo zajistit způsob zobrazení změn osob a parcel s věcným právem či břemenem dotčených pozemkovou úpravou. Pro výběr informací byl použit dotazovací jazyk SQL.

Všechny výsledné dotazy jsou uvedeny v příloze A a jsou to: výpis povinných osob, výpis oprávněných osob a výpis parcel s věcným břemenem. Všechny dotazy mají identický začátek, který se ukládá do pomocné tabulky a to pro zrychlení práce s nimi. Stačí potom první část spustit jen při prvním dotazu, ostatní pak mohou vybírat data už z této předpřipravené tabulky.

Napsané dotazy pro výpis změn osob povinných a oprávněných byly vloženy do zdrojového kódu programu PROLAND, otestovány a od května (od verze 9.33) jsou již součástí nové distribuční verze programu. Výpis změn parcel s věcným břemenem zatím součástí distribuční verze není, ale jeho zapracování je plánováno do některé z příštích verzí.

Toto rozšíření jistě pomůže projektantům pozemkových úprav k lepší orientaci ve změnách, které se v obvodu pozemkových úprav průběžně dějí. Při práci na tomto rozšíření se objevily možnosti dalšího vylepšování programu PROLAND týkající se práce s věcnými břemeny. Jde především o funkce pro změnu, rušení a zadávání nových věcných břemen při projektování pozemkových úprav.

Použité zdroje

[1] Zákon 139/2002 Sb. *O pozemkových úpravách a pozemkových úřadech.*

[2] Vyhláška 545/2002 Sb. *O postupu při provádění pozemkových úprav a náležitostech návrhu pozemkových úprav.*

[3] JANOVSKÝ, F. *Pozemkové úpravy* , Zeměměřič, 12/2004.

[4] VLASÁK, J. - BARTOŠKOVÁ, H. *Pozemkové úpravy* , Praha, ČVUT, 2007.

[5] Zákon 40/1964 Sb. *Občanský zákoník.*

[6] OPPEL, A. *SQL bez předchozích znalostí* , 2008.

[7] SAGIT *VĚCNÉ BŘEMENO - Občanské právo - Sagit* [online]. Poslední aktualizace 15. 5. 2009. Dostupné z URL: <http://www.sagit.cz>.

[8] Právník.cz *Věcné břemeno - Právník.cz* [online]. Poslední aktualizace 15. 5. 2009. Dostupné z URL: <http://www.pravnik.cz/a/187/vecne-bremeno.html>.

[9] GEPRO spol. s r.o. *PROLAND - Uživatelská příručka* 2008.

Seznam symbolů, veličin a zkratek

SPI	Soubor popisných informací
PÚ	Pozemkové úpravy
KN	Katastr nemovitostí
PK	Pozemkový katastr
LV	List vlastnictví
SJM	Společné jmění manželů
BSM	Bezpodílové spoluvlastnictví manželů
VFK	Výměnný formát katastru nemovitostí
BPEJ	Bonitovaná půdně ekologická jednotka

Seznam příloh

A SQL dotazy

A.1 Výpis pro povinné osoby

```
drop table [E:\zalany\pokus\pokus1.mdb].tmp_par

SELECT pkn.katuze, pkn.parskup, pkn.parcis, pkn.parpod, pkn.pardil,
pkn.cislo_lv, pkn.par_id, max(cp.zajem_upr) as ZU
INTO [E:\zalany\pokus\pokus1.mdb].tmp_par
FROM (parcely_kn pkn inner join KU on (KU.katuze=pkn.katuze))
inner join casti_parcel cp on (pkn.id = cp.parcela_kn)
WHERE EXISTS (SELECT 'i' FROM casti_parcel cp
WHERE cp.parcela_kn = pkn.id AND cp.zajem_upr>0)
group by pkn.katuze, pkn.parskup, pkn.parcis, pkn.parpod, pkn.pardil,
pkn.cislo_lv, pkn.par_id

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_par
SELECT ppk.katuze, ppk.parskup, ppk.parcis, ppk.parpod, ppk.pardil,
ppk.cislo_lv, ppk.par_id, max(cp.zajem_upr) as ZU
FROM (parcely_pk ppk inner join KU on (KU.katuze = ppk.katuze) )
inner join casti_parcel cp on (ppk.id = cp.parcela_pk)
WHERE EXISTS (SELECT 'i' FROM casti_parcel cp
inner join parcely_kn k ON (cp.parcela_kn = k.id)
WHERE cp.parcela_pk = ppk.id AND cp.zajem_upr>0
AND (k.cislo_lv<1 OR k.cislo_lv IS NULL))
group by ppk.katuze, ppk.parskup, ppk.parcis, ppk.parpod, ppk.pardil,
ppk.cislo_lv, ppk.par_id

create index PAR_ID on [E:\zalany\pokus\pokus1.mdb].tmp_par (PAR_ID)

drop table [E:\zalany\pokus\pokus1.mdb].tmp_bud
```

```
SELECT DISTINCT k.bud_id, t.katuze
INTO [E:\zalany\pokus\pokus1.mdb].tmp_bud
FROM parcely_kn k WHERE EXISTS (SELECT 'i'
FROM [E:\zalany\pokus\pokus1.mdb].tmp_par t
WHERE (t.par_id = k.par_id)) AND k.bud_id IS NOT NULL

create index BUD_ID on [E:\zalany\pokus\pokus1.mdb].tmp_bud (BUD_ID)

drop table [E:\zalany\pokus\pokus1.mdb].tmp_jpv

SELECT DISTINCT j.opsub_id_k AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
FROM jine_prav_vztahy j
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_k = t.par_id)
WHERE j.opsub_id_pro IS NOT NULL

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT j.opsub_id_k AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID,
j.typrav_kod FROM jine_prav_vztahy j
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro = t.par_id)
WHERE j.opsub_id_k IS NOT NULL

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_kn k
```

```
on (k.cislo_lv = v.cislo_lv) AND (k.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_k = k.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro=t.par_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_pk p
on (p.cislo_lv = v.cislo_lv) AND (p.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_k = p.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro=t.par_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v
inner join budovy b on (b.cislo_tel = v.cislo_lv)
AND (b.katuze_kod = v.katuze))
inner join jine_prav_vztahy j on (j.bud_id_k = b.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro = t.par_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 16 as TYP,
j.par_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v
inner join jednotky jed on (jed.cislo_lv = v.cislo_lv)
AND (jed.katuze=v.katuze))
inner join jine_prav_vztahy j on (j.jed_id_k = jed.id))
```

```
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_pro=t.par_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT j.opsub_id_k AS ID, t.katuze, 32 AS TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
FROM jine_prav_vztahy j
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id)
WHERE j.opsub_id_k IS NOT NULL

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 32 as TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod F
ROM ((vlastnictvi v
inner join parcely_kn k on (k.cislo_lv = v.cislo_lv)
AND (k.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_k = k.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 32 as TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v
inner join parcely_pk p on (p.cislo_lv = v.cislo_lv)
AND (p.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_k = p.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id)
```

```
INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 32 as TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v
inner join budovy b on (b.cislo_tel = v.cislo_lv)
AND (b.katuze_kod = v.katuze))
inner join jine_prav_vztahy j on (j.bud_id_k = b.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 32 as TYP,
j.bud_id_pro AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v
inner join jednotky jed on (jed.cislo_lv = v.cislo_lv)
AND (jed.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.jed_id_k = jed.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_pro = t.bud_id)

create index ID on [E:\zalany\pokus\pokus1.mdb].tmp_jpv (ID)

drop table [E:\zalany\pokus\pokus1.mdb].tmp_op

SELECT j.id, j.katuze, j.typ, j.typrav_kod, j.obj_id
into [E:\zalany\pokus\pokus1.mdb].tmp_op
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.id = o.id)
WHERE o.opsub_type<>'BSM'

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_op
```

```
SELECT o.id_je_1_partner_bsm, j.katuze, j.typ, j.typrav_kod,
j.obj_id
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.ID = o.id)
WHERE o.opsub_type = 'BSM'

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_op
SELECT o.id_je_2_partner_bsm, j.katuze, j.typ, j.typrav_kod, j.obj_id
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.ID = o.id)
WHERE o.opsub_type = 'BSM'

SELECT DISTINCT o.id, t.katuze, o.opsub_type, o.ico, o.doplnek_ico,
o.nazev, o.rodne_cislo, o.titul_pred_jmenem, o.jmeno, o.prijmeni,
o.titul_za_jmenem, o.cislo_domovni, o.cislo_orientacni, o.nazev_ulice,
o.cast_obce, o.mestska_cast, o.obec, o.okres, o.stat, o.psc, t.typ
FROM [E:\zalany\pokus\pokus1.mdb].tmp_op t
inner join oprav_subjekty o ON (t.id = o.id)
ORDER BY o.id
```

A.2 Výpis pro jiné oprávněné

```
drop table [E:\zalany\pokus\pokus1.mdb].tmp_par

SELECT pkn.katuze, pkn.parskup, pkn.parcis, pkn.parpod, pkn.pardil,
pkn.cislo_lv, pkn.par_id, max(cp.zajem_upr) as ZU
INTO [E:\zalany\pokus\pokus1.mdb].tmp_par
FROM (parcely_kn pkn inner join KU on (KU.katuze=pkn.katuze))
inner join casti_parcel cp on (pkn.id = cp.parcela_kn)
WHERE EXISTS (SELECT 'i' FROM casti_parcel cp
WHERE cp.parcela_kn = pkn.id AND cp.zajem_upr > 0)
group by pkn.katuze, pkn.parskup, pkn.parcis, pkn.parpod, pkn.pardil,
pkn.cislo_lv, pkn.par_id

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_par
SELECT ppk.katuze, ppk.parskup, ppk.parcis, ppk.parpod, ppk.pardil,
ppk.cislo_lv, ppk.par_id, max(cp.zajem_upr) as ZU
FROM (parcely_pk ppk inner join KU on (KU.katuze = ppk.katuze) )
inner join casti_parcel cp on (ppk.id = cp.parcela_pk)
WHERE EXISTS (SELECT 'i' FROM casti_parcel cp
INNER JOIN parcely_kn k ON (cp.parcela_kn = k.id)
WHERE cp.parcela_pk = ppk.id AND cp.zajem_upr > 0
AND (k.cislo_lv < 1 OR k.cislo_lv IS NULL))
group by ppk.katuze, ppk.parskup, ppk.parcis, ppk.parpod, ppk.pardil,
ppk.cislo_lv, ppk.par_id

create index PAR_ID on [E:\zalany\pokus\pokus1.mdb].tmp_par (PAR_ID)

drop table [E:\zalany\pokus\pokus1.mdb].tmp_bud

SELECT DISTINCT k.bud_id, k.katuze
```

```
INTO [E:\zalany\pokus\pokus1.mdb].tmp_bud
FROM parcely_kn k WHERE EXISTS (SELECT 'i'
FROM [E:\zalany\pokus\pokus1.mdb].tmp_par t
WHERE (t.par_id = k.par_id)) AND k.bud_id IS NOT NULL

create index BUD_ID on [E:\zalany\pokus\pokus1.mdb].tmp_bud (BUD_ID)

drop table [E:\zalany\pokus\pokus1.mdb].tmp_jpv

SELECT DISTINCT j.opsub_id_pro AS ID, t.katuze, 4 as TYP,
j.par_id_k AS OBJ_ID, j.typrav_kod
INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
FROM jine_prav_vztahy j
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t
on (j.par_id_k=t.par_id) WHERE j.opsub_id_pro IS NOT NULL

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv S
ELECT DISTINCT v.opsub_id AS ID,t.katuze, 4 as TYP,
j.par_id_k AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_kn k
on (k.cislo_lv = v.cislo_lv) AND (k.katuze=v.katuze))
inner join jine_prav_vztahy j on (j.par_id_pro = k.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t on
(j.par_id_k = t.par_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 4 as TYP,
j.par_id_k AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_pk p
on (p.cislo_lv = v.cislo_lv) AND (p.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_pro = p.par_id))
```

```
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t on
(j.par_id_k = t.par_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 4 as TYP,
j.par_id_k AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join budovy b
on (b.cislo_tel = v.cislo_lv) AND (b.katuze_kod = v.katuze))
inner join jine_prav_vztahy j on (j.bud_id_pro = b.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t on
(j.par_id_k = t.par_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 4 as TYP,
j.par_id_k AS OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join jednotky jed
on (jed.cislo_lv = v.cislo_lv) AND (jed.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.jed_id_pro = jed.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_par t on
(j.par_id_k = t.par_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT j.opsub_id_pro AS ID, t.katuze, 8 AS TYP,
j.bud_id_k AS OBJ_ID, j.typrav_kod
FROM jine_prav_vztahy j
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_k = t.bud_id)
WHERE j.opsub_id_pro IS NOT NULL

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 8 as TYP,
```

```
j.bud_id_k as OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_kn k
on (k.cislo_lv = v.cislo_lv) AND (k.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_pro = k.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_k = t.bud_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 8 as TYP,
j.bud_id_k as OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join parcely_pk p
on (p.cislo_lv = v.cislo_lv) AND (p.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.par_id_pro = p.par_id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_k = t.bud_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 8 as TYP,
j.bud_id_k as OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join budovy b
on (b.cislo_tel = v.cislo_lv) AND (b.katuze_kod = v.katuze))
inner join jine_prav_vztahy j on (j.bud_id_pro = b.id))
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t
on (j.bud_id_k = t.bud_id)

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_jpv
SELECT DISTINCT v.opsub_id AS ID, t.katuze, 8 as TYP,
j.bud_id_k as OBJ_ID, j.typrav_kod
FROM ((vlastnictvi v inner join jednotky jed
on (jed.cislo_lv = v.cislo_lv) AND (jed.katuze = v.katuze))
inner join jine_prav_vztahy j on (j.jed_id_pro = jed.id))
```

```
inner join [E:\zalany\pokus\pokus1.mdb].tmp_bud t on
(j.bud_id_k = t.bud_id)

create index ID on [E:\zalany\pokus\pokus1.mdb].tmp_jpv (ID)

drop table [E:\zalany\pokus\pokus1.mdb].tmp_op

SELECT j.id, j.katuze, j.typ, j.typrav_kod,
j.obj_id into [E:\zalany\pokus\pokus1.mdb].tmp_op
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.id = o.id)
WHERE o.opsub_type <> 'BSM'

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_op
SELECT o.id_je_1_partner_bsm, j.katuze, j.typ, j.typrav_kod, j.obj_id
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.ID = o.id)
WHERE o.opsub_type = 'BSM'

INSERT INTO [E:\zalany\pokus\pokus1.mdb].tmp_op
SELECT o.id_je_2_partner_bsm, j.katuze,j.typ, j.typrav_kod, j.obj_id
FROM [E:\zalany\pokus\pokus1.mdb].tmp_jpv j
inner join oprav_subjekty o ON (j.ID = o.id) WHERE o.opsub_type = 'BSM'

SELECT DISTINCT o.id, t.katuze, o.opsub_type, o.ico, o.doplnek_ico,
o.nazev, o.rodne_cislo, o.titul_pred_jmenem, o.jmeno, o.prijmeni,
o.titul_za_jmenem, o.cislo_domovni, o.cislo_orientacni, o.nazev_ulice,
o.cast_obce, o.mestska_cast, o.obec, o.okres, o.stat, o.psc, t.typ
FROM [E:\zalany\pokus\pokus1.mdb].tmp_op t
inner join oprav_subjekty o ON (t.id = o.id)
ORDER BY o.id
```

A.3 Výpis parcel s věcným břemenem

```
drop table [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_par

SELECT pkn.katuze, pkn.parskup, pkn.parcis, pkn.parpod, pkn.pardil,
pkn.cislo_lv, pkn.par_id, max(cp.zajem_upr) as ZU
INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_par
FROM (parcely_kn pkn inner join KU on (KU.katuze = pkn.katuze))
inner join casti_parcel cp on (pkn.id = cp.parcela_kn)
WHERE EXISTS (SELECT 'i' FROM casti_parcel cp
WHERE cp.parcela_kn = pkn.id AND cp.zajem_upr > 0)
group by pkn.katuze, pkn.parskup, pkn.parcis, pkn.parpod,
pkn.pardil, pkn.cislo_lv, pkn.par_id

INSERT INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_par
SELECT ppk.katuze, ppk.parskup, ppk.parcis, ppk.parpod, ppk.pardil,
ppk.cislo_lv, ppk.par_id, max(cp.zajem_upr) as ZU
FROM (parcely_pk ppk inner join KU on (KU.katuze = ppk.katuze) )
inner join casti_parcel cp on (ppk.id = cp.parcela_pk)
WHERE EXISTS (SELECT 'i' FROM casti_parcel cp
INNER JOIN parcely_kn k ON (cp.parcela_kn = k.id)
WHERE cp.parcela_pk = ppk.id AND cp.zajem_upr > 0
AND (k.cislo_lv < 1 OR k.cislo_lv IS NULL))
group by ppk.katuze, ppk.parskup, ppk.parcis, ppk.parpod,
ppk.pardil, ppk.cislo_lv, ppk.par_id

create index PAR_ID
on [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_par (PAR_ID)

drop table [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2
```

```
SELECT j.PAR_ID_K, j.TYPRAV_KOD, j.POPIS_PRAV_VZTAHU, 1 as TYP,
b.ID as OID, b.KATUZE_KOD as KK, b.CISLO_DOMOVNI as PS, b.TYPBUD_KOD as PC,
0 as PP, 0 as PD, 0 as KP
INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2
FROM JINE_PRAV_VZTAHY j inner join BUDOVY b on (b.ID = j.BUD_ID_PRO)
WHERE (j.PAR_ID_K is not NULL)

INSERT INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2
SELECT j.PAR_ID_K, j.TYPRAV_KOD, j.POPIS_PRAV_VZTAHU, 2 as TYP,
p.PAR_ID as OID, p.KATUZE as KK, p.PARSKUP as PS, p.PARCIS as PC,
p.PARPOD as PP, p.PARDIL as PD,0  as KP
FROM JINE_PRAV_VZTAHY j
inner join PARCELY_KN p on (p.PAR_ID=j.PAR_ID_PRO)
WHERE (j.PAR_ID_K is not NULL)

INSERT INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2
SELECT j.PAR_ID_K, j.TYPRAV_KOD, j.POPIS_PRAV_VZTAHU, 2 as TYP,
p.PAR_ID as OID, p.KATUZE as KK, p.PARSKUP as PS,p.PARCIS as PC,
p.PARPOD as PP, p.PARDIL as PD, p.KATUZE_PUV as KP f
FROM JINE_PRAV_VZTAHY j
 inner join PARCELY_PK p on (p.PAR_ID = j.PAR_ID_PRO)
WHERE (j.PAR_ID_K is not NULL)

INSERT INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2
SELECT j.PAR_ID_K, j.TYPRAV_KOD, j.POPIS_PRAV_VZTAHU, 3 as TYP,
j.JED_ID_PRO as OID, jed.KATUZE as KK, jed.CISLO_DOMOVNI as PS,
jed.CISLO_JED as PC, 0 as PP, 0 as PD, 0 as KP
FROM JINE_PRAV_VZTAHY j
inner join JEDNOTKY jed on (jed.ID = j.JED_ID_PRO)
WHERE (j.PAR_ID_K is not NULL)
```

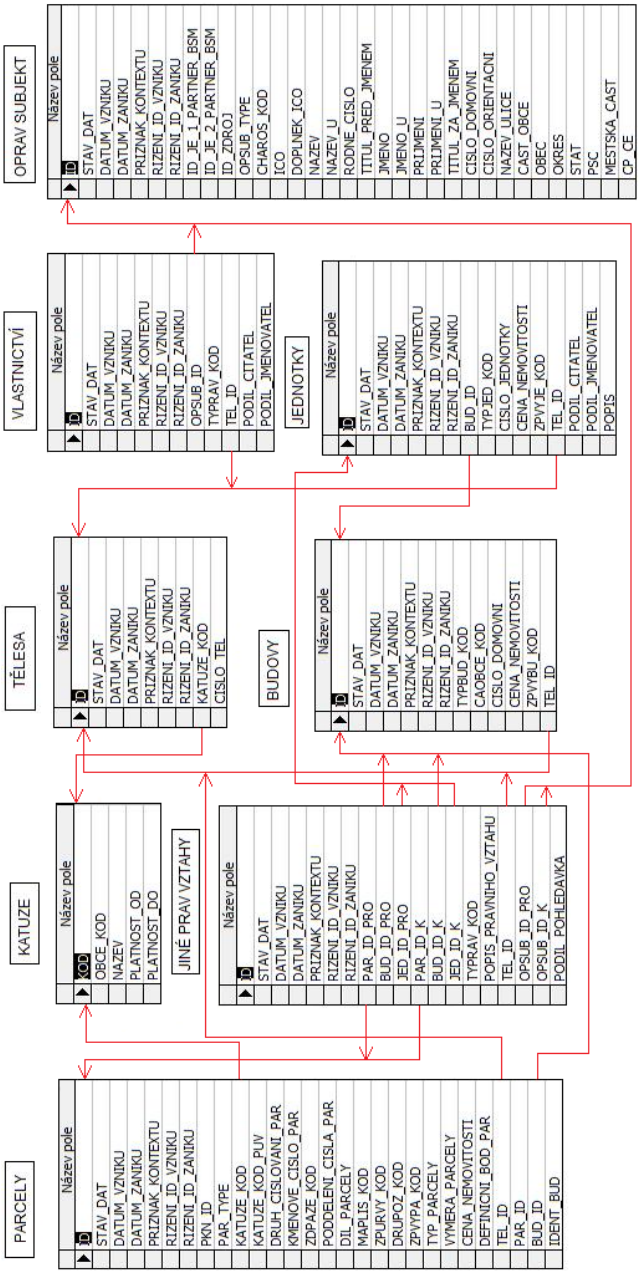
```
INSERT INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2
SELECT j.PAR_ID_K, j.TYPRAV_KOD, j.POPIS_PRAV_VZTAHU, 4 as TYP,
j.OPSUB_ID_PRO as OID, 0 as KK, 0 as PS, 0 as PC, 0 as PP,
0 as PD, 0 as KP
FROM JINE_PRAV_VZTAHY j
WHERE (j.OPSUB_ID_PRO is not NULL) and (j.PAR_ID_K is not NULL)

INSERT INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2
SELECT j.PAR_ID_K, j.TYPRAV_KOD, j.POPIS_PRAV_VZTAHU, 5 as TYP,
NULL as OID, 0 as KK, 0 as PS, 0 as PC, 0 as PP, 0 as PD, 0 as KP
FROM JINE_PRAV_VZTAHY j
WHERE (j.OPSUB_ID_PRO is NULL) AND (j.OPSUB_ID_K is NULL)
AND (j.PAR_ID_PRO is NULL) AND (j.BUD_ID_PRO is NULL)
AND (j.JED_ID_PRO is NULL) AND (j.PAR_ID_K is not NULL)

INSERT INTO [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2
SELECT j.PAR_ID_K, j.TYPRAV_KOD, j.POPIS_PRAV_VZTAHU, 6 as TYP,
j.OPSUB_ID_K as OID, 0 as KK, 0 as PS, 0 as PC, 0 as PP, 0 as PD,
0 as KP
FROM JINE_PRAV_VZTAHY j
WHERE (j.OPSUB_ID_K is not NULL) AND (j.PAR_ID_K is not NULL)

SELECT a.PAR_ID, a.KATUZE, a.PARSKUP, a.PARCIS, a.PARPOD,
a.PARDIL, a.CISLO_LV, b.kk, b.TYPRAV_KOD, b.PS, b.TYP, b.PC,
b.PP, b.PD, b.KP, b.POPIS_PRAV_VZTAHU, b.OID
FROM [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_par a
inner join [E:\data_vse\databaze(zalany)\pokus\pokus1.mdb].tmp_parc2 b
on (a.PAR_ID = b.PAR_ID_K)
ORDER BY a.KATUZE, a.PARSKUP, a.PARCIS, a.PARPOD, a.PARDIL, b.TYPRAV_KOD
```

B Tabulky



Obr. B.1: Tabulky