EKONOMICKÁ FAKULTA TUL

**Diplomová práce**

# Analýza adresních dat z OpenStreetMap a jejich automatizovaná transformace do platformy Grasshopper


| | |
|---|---|
| *Studijní program:* | N0688A140016 Systémové inženýrství a informatika |
| *Autor práce:* | **Bc. Jan Míka** |
| *Vedoucí práce:* | Ing. Athanasios Podaras, Ph.D.<br>Katedra informatiky |


Liberec 2024

EKONOMICKÁ FAKULTA TUL

**Zadání diplomové práce**

# Analýza adresních dat z OpenStreetMap a jejich automatizovaná transformace do platformy Grasshopper

| | |
|---|---|
| *Jméno a příjmení:* | **Bc. Jan Míka** |
| *Osobní číslo:* | E22000620 |
| *Studijní program:* | N0688A140016 Systémové inženýrství a informatika |
| *Zadávající katedra:* | Katedra informatiky |
| *Akademický rok:* | 2023/2024 |

**Zásady pro vypracování:**

1. Cíle práce, důvod zavedení nástroje
2. ETL proces OpenStreetMap, Grasshopper a využívané platformy
3. Datové modely, databáze PostgreSQL, programovací jazyky potřebné k vývoji nástroje
4. Vytvoření nástroje pro transformaci dat a jejich analýzu
5. Popis přínosů, diskuze a formulace závěrů

*Rozsah grafických prací:*
*Rozsah pracovní zprávy:* 65 normostran
*Forma zpracování práce:* tištěná/elektronická
*Jazyk práce:* čeština

## Seznam odborné literatury:


- BARRON, Christopher; Pascal NEIS a Alexander ZIPF, 2013. A Comprehensive Framework for Intrinsic OpenStreetMap Quality Analysis. *Transactions in GIS*, vol. 18, no. 6, s. 877–895. ISSN:1467-9671.
- FERRARI, Luca a Enrico PIROZZI, 2020. *Learn PostgreSQL*. Birmingham – Mumbai: Packt Publishing. ISBN: 978-1-83898-528-8.
- PECINOVSKÝ, Rudolf, 2022. *Python – knihovny pro práci s daty*. Česko: Grada. ISBN 978-80-271-0659-2.
- SAKR, Sherif, 2020. *Big Data 2.0 Processing Systems: a Systems Overview*. Cham, Switzerland: Springer Nature. ISBN: 978-3-030-44186-9.


Konzultant: Ing. Jan Herold, Trixi software s.r.o

*Vedoucí práce:* Ing. Athanasios Podaras, Ph.D.
Katedra informatiky

*Datum zadání práce:* 1. listopadu 2023
*Předpokládaný termín odevzdání:* 31. srpna 2025

L.S.

doc. Ing. Aleš Kocourek, Ph.D.
děkan

doc. Ing. Klára Antlová, Ph.D.
garant studijního programu

V Liberci dne 1. listopadu 2023

# Prohlášení

Prohlašuji, že svou diplomovou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Jsem si vědom toho, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má diplomová práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

2. května 2024 Bc. Jan Míka

## Anotace

Tato diplomová práce se zabývá analýzou adresních dat z OpenStreetMap a jejich automatizovanou transformací do platformy Grasshopper. Teoretická část práce se zaměřuje na popis potřebných nástrojů pro analýzu dat, jako jsou ETL proces OpenStreetMap, Grasshopper a využívané platformy, datové modely, databáze a programovací jazyky potřebné k vývoji nástroje. Praktická část práce se zabývá implementací nástroje pro transformaci dat a jejich analýzu. Výsledky této práce ukazují, že vytvořený nástroj umožňuje automatizovanou transformaci dat z OpenStreetMap do platformy Grasshopper a jejich analýzu. Závěrem lze konstatovat, že vytvořený nástroj přináší nové možnosti pro analýzu adresních dat z OpenStreetMap.

## Klíčová slova

Grasshopper, OpenStreetMap, ETL, PostgreSQL, Java, Kotlin, Automatizovaná transformace

## Annotation

**Analysis of address data from OpenStreetMap and its automated transformation into the Grasshopper platform**

This thesis deals with the analysis of address data from OpenStreetMap and its automated transformation into the Grasshopper platform. The theoretical part of the thesis focuses on describing the tools needed for data analysis, such as the ETL process of OpenStreetMap, Grasshopper and the platforms used, data models, databases and programming languages needed to develop the tool. The practical part of the thesis deals with the implementation of the data transformation and it’s analysis. The results of this thesis show that the developed tool enables automated data transformation from OpenStreetMap to the Grasshopper platform and its analysis. In conclusion, the developed tool brings new possibilities for the analysis of address data from OpenStreetMap.



## Key words

Grasshopper, OpenStreetMap, ETL, PostgreSQL, Java, Kotlin, Automated transformation

**Poděkování**

Na tomto místě bych chtěl vyjádřit své upřímné díky vedoucímu mé diplomové práce, Ing. A. Podarasovi, Ph.D., za jeho pomoc, ochotu a profesionální vedení mé práce.

Speciální poděkování patří také konzultantovi mé práce, panu Ing. Janu Heroldovi, za jeho čas a cenné odborné znalosti, které mi předal při tvorbě systému pro práci s daty OpenStreetMap.

Nemohu opomenout poděkovat i společnosti Trixi, která mi poskytla příležitost vypracovat svou diplomovou práci v jejich prostředí. Jejich podpora a otevřenost byla klíčová pro její úspěšné dokončení.

# Obsah

# Seznam obrázků

# Seznam použitých zkratek

| | |
|---|---|
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| CPU | Central Processing Unit |
| DBMS | Database Management System |
| ETL | Extract, Transform, Load |
| GIS | Geographic Information Systems |
| GPS | Global Positioning System |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Hypertext Transfer Protocol Secure |
| ID | Identification |
| JSON | JavaScript Object Notation |
| JTSK | Jednotný trigonometrický katastrální síť |
| OSM | OpenStreetMap |
| PBF | Protocolbuffer Binary Format |
| RAM | Random Access Memory |
| SQL | Structured Query Language |
| SSD | Solid-State Drive |
| UI | User Interface |
| URL | Uniform Resource Locator |
| WGS 84 | World Geodetic System 1984 |
| XML | eXtensible Markup Language |

# Úvod

V posledních letech se význam geografických informačních systémů (GIS) a prostorových dat v různých oblastech lidské činnosti neustále zvyšuje. S rostoucím objemem dostupných prostorových dat a jejich aplikací v urbanismu, dopravě, environmentálních studiích a mnoha dalších oborech se zvyšují i požadavky na efektivní nástroje pro jejich analýzu a vizualizaci. V této souvislosti se projekt OpenStreetMap (OSM) ukazuje jako cenný zdroj volně dostupných prostorových dat s širokou škálou využití.

Tato diplomová práce se zabývá analýzou a automatizovanou transformací adresních dat z OpenStreetMap do platformy Grasshopper. Cílem práce je navrhnout a implementovat nástroj, který umožní efektivní přenos a analýzu těchto adresních dat, a tím otevřít nové možnosti pro jejich využití v urbanistickém designu, plánování a analýze. Práce kombinuje teoretické poznatky v oblasti GIS, databázových systémů a programování s praktickou implementací nástroje, který reflektuje současné potřeby a trendy v oblasti prostorového plánování a designu.

V úvodní části se práce věnuje představení problematiky a významu adresních dat v kontextu prostorové analýzy. Dále jsou zde objasněny klíčové koncepty a technologie, které hrají roli v procesu transformace a analýzy dat, včetně podrobného přehledu o OpenStreetMap, Grasshopperu, metodách ETL (Extract, Transform, Load) a relevantních programovacích jazycích. V následujících kapitolách je podrobně rozebrán návrh a implementace nástroje, včetně výběru technologií, architektury řešení a praktické demonstrace jeho použití na vybraných příkladech. Závěrečná část práce pak hodnotí dosažené výsledky, diskutuje potenciální aplikace a možnosti dalšího rozvoje nástroje.

Cílem této práce je tedy nejen představit nový nástroj pro analýzu a transformaci adresních dat, ale také prohloubit pochopení možností integrace různých datových zdrojů a analytických nástrojů v oblasti prostorového plánování a designu. Přináší tak nový pohled na využití otevřených dat a softwarových nástrojů v praxi a otevírá diskusi o budoucím směřování a potenciálu těchto technologií ve stále se vyvíjejícím poli urbanismu a architektury.

Pro přípravu textu této diplomové práce bylo využito uplatnění umělé inteligence, konkrétně Chat GPT 4.0. Využití umělé inteligence probíhalo sporadicky a přerušovaně, avšak téměř v celém rozsahu práce.

# 1 Geografická data

Geodata, známá také jako geografická data, jsou informace, které lze přiřadit k určitému místu na povrchu Země. Tato data mají klíčový význam v mnoha odvětvích a aplikacích, což je zřejmé z růstu hodnoty lokalizačních služeb, který byl zaznamenán v posledních letech. Jak je zdůrazněno v knize "Geographic Information Systems and Science", geodata jsou zásadní pro rozhodovací procesy v širokém spektru oblastí, od urbanistického plánování až po environmentální management. Rychlý rozvoj technologií, jako jsou mobilní zařízení, digitální sběr dat a crowdsourcing, značně rozšířil možnosti získávání a využívání geodat. Dále, integrace těchto dat do geografických informačních systémů (GIS) umožňuje složité analýzy, vizualizaci a interpretaci prostorových vzorců a trendů, což je nezbytné pro pochopení a řešení komplexních geografických a sociálních výzev. Tato data tak hrají nezastupitelnou roli v moderním světě, kde se významně podílejí na rozvoji a inovacích ve všech sektorech společnosti. (Paul A. Longley, 2015)

Geografická data hrají klíčovou roli ve zpracování informací, zejména v kontextu analýzy adres a prostorového plánování. Otevřená data poskytují cenný zdroj pro geografické analýzy, a v tomto rámci zaujímá OpenStreetMap (OSM) významné místo.

## 1.1 Definice a význam

Geografická data jsou souborem informací, které kombinují prostorové (lokální) a deskriptivní (popisné) atributy k identifikaci, umístění a charakterizaci objektů, jevů a hranic v geografickém kontextu. Tato data jsou obvykle reprezentována pomocí souřadnicových systémů a mohou zahrnovat různé typy informací, jako jsou fyzické vlastnosti (například terén, vodstvo), infrastrukturální prvky (například cesty, budovy) nebo demografické údaje (například hustota obyvatelstva, rozložení populace). Geodata jsou klíčová pro geografické informační systémy (GIS), kde slouží k analýze, vizualizaci a porozumění prostorovým vzorcům a vztahům.

Význam geodat lze chápat z několika perspektiv:

**Modelování reality**: Geodata slouží k modelování reality, obsahují údaje o objektech a jevech v území, včetně jejich polohy, tvaru a vlastností.

**Datové modely v GIS:** Míra, jakou geodata popisují vzájemné vazby, závisí na datovém modelu. Datový model GIS je popisný model reality a je vytvářen tak, aby vyhovoval předpokládané aplikaci.

**Sdílení geodat:** Sdílení geodat zahrnuje publikování pro vizualizaci, poskytování dat umožňující změnu datového modelu pro nového uživatele a poskytování specializovaných služeb, jako jsou geoprocessingové služby, image služby, network služby a další.

**Nákladnost a efektivita:** Geodata jsou považována za nejnákladnější část GIS. Důležité je zvážit, jaká geodata jsou v aplikaci potřebná, hledat efektivní metody tvorby geodat, sdílet geodata, snižovat náklady na poskytování geodat a zefektivnit jejich využití.

(Seidl, 2003)

## 1.2 Typy a formáty

**Data:** Toto jsou nejjednodušší a nejzákladnější prvky informací. Jsou to čísla, texty nebo symboly, které jsou samy o sobě neutrální a téměř bez kontextu. Příkladem jsou měření teploty senzorem v určitém čase a na určitém místě. Data jsou jako surová fakta a při přenosu se zacházejí jako proud bitů, kde je důležitá integrita datové sady, ale její vnitřní význam není v tuto chvíli relevantní. (Paul A. Longley, 2015)

**Informace:** Informace se od dat liší tím, že zahrnují určitou míru výběru, organizace a přípravy pro konkrétní účely. Informace jsou data sloužící nějakému účelu nebo data, která byla nějakým způsobem interpretována. Například geografické datové sady mohou být velmi nákladné na shromažďování, ale jednou digitalizované jsou levné na kopírování a šíření. Informace lze snadno zvýšit hodnotu jejich zpracováním a sloučením s dalšími informacemi. (Paul A. Longley, 2015)

**Důkazy (Evidence):** Důkazy jsou považovány za "přechodný stav" mezi informacemi a znalostmi. Jsou to mnohonásobné informace z různých zdrojů, související s konkrétními problémy a mající určitou konzistenci, která byla ověřena. Důkazy vznikají ze souboru informací, které byly validovány a jsou relevantní pro určité problémy. (Paul A. Longley, 2015)

**Znalosti (Knowledge):** Znalost není jen o přístupu k velkému množství informací, ale zahrnuje i hodnotu přidanou interpretací založenou na konkrétním kontextu, zkušenostech a účelu. Znalosti se stávají z informací, když jsou čteny a pochopeny. Existují dva typy znalostí: kodifikované, které lze snadno zapsat a předat, a tacitní, které jsou těžší získat a přenášet. (Paul A. Longley, 2015)

**Moudrost (Wisdom):** Moudrost je nejvíce neuchopitelný pojem. Obvykle se používá ve smyslu rozhodnutí nebo rad, které jsou nezainteresované, založené na veškerých

dostupných důkazech a znalostech a s porozuměním pravděpodobných důsledků různých akcí. Moudrost je na nejvyšší úrovni hierarchie rozhodovací infrastruktury a je velmi individualizovaná.

(Paul A. Longley, 2015)

Data ve většině případů obsahují tři základní informace, těmi jsou:
**Prostorová informace:** Tento typ informace se věnuje poloze objektu a jeho tvaru.
**Popisná informace:** Zahrnuje další vlastnosti objektu, které jej popisují, jako jsou název, adresa, teplota, tloušťka drátu a podobně.
**Časová informace:** Když se využívá časová informace, přidává to systému dynamickou vlastnost a umožňuje tak sledování změn v čase.
Dále je tyto tři informace možno zobrazovat digitálně v geografických informačních systémech nebo analogově na mapě.

(Jedlička, 2022)

Podle norem ISO/OGC existuje devět metod pro testování prostorových vztahů mezi geometrickými objekty, například vzdálenost, nárazník, konvexní trup, průnik a sjednocení. Tyto metody vyhodnocují vztahy mezi dvěma geometriemi (kolekcemi jednoho nebo více geometrických objektů). (Paul A. Longley, 2015)

### 1.2.1 Sběr dat

Sběr dat je jednou z nejnáročnějších a nejdražších, ale zároveň nejdůležitějších úloh spojených s geografickými informacemi (GI). Existuje mnoho rozmanitých zdrojů GI a mnoho metod pro jejich vkládání do GI systémů. Dvě hlavní metody sběru dat jsou zachytávání dat a přenos dat. Užitečné je rozlišovat mezi primárním (přímé měření) a sekundárním (odvození z jiných zdrojů) zachytáváním dat pro oba typy dat – rastrová a vektorová. Přenos dat zahrnuje import digitálních dat z jiných zdrojů, jako jsou geoportály. Plánování a provádění efektivního plánu sběru dat zahrnuje mnoho praktických aspektů. Tato kapitola přehledně popisuje hlavní metody zachytávání a přenosu dat a uvádí klíčové praktické aspekty managementu. (Paul A. Longley, 2015)

### 1.2.2 Strukturování a organizace geografických databází

Dvěma hlavními technikami strukturování jsou vytváření topologie a indexování. Topologie se zaměřuje na strukturování a ukládání v DBMS systému GI, přičemž existují dva hlavní přístupy: normalizovaný a fyzický. Normalizovaný model ukládá každý objekt jako

jednotlivé topologické primitivy pro opětovné sestavení při dotazech. Fyzický model ukládá celou geometrii pro každý objekt a počítá topologické vztahy za běhu.

**Indexování v geografických databázích:** Indexování se používá k urychlení dotazů ve velkých geografických databázích tím, že snižuje počet výpočetních testů potřebných k nalezení souboru záznamů. Tím se zabrání nákladnému prohledávání celých tabulek. Databázový index je speciální reprezentace informací o objektech, která zlepšuje vyhledávání. (Paul A. Longley, 2015)

## 1.3 Georeferencing

Georeferencování je klíčovým procesem v geografických informačních systémech (GIS), který spočívá v přiřazení geografických dat k určitém bodům na zemském povrchu. Tento proces umožňuje, aby se data shromážděná z různých zdrojů dala efektivně použít, interpretovat a vizualizovat v kontextu reálného světa. Přesnost georeferencování ovlivňuje, jak přesně můžeme lokalizovat objekty a jevy na Zemi, což má zásadní význam pro širokou škálu aplikací, od urbanistického plánování po navigaci a vědecký výzkum.

Techniky georeferencování se vyvíjely po staletí, ale v posledních letech se staly přístupnější a přesnější díky pokrokům v GIS a web-based službách. Georeferencování zahrnuje různé metody, od jednoduchých, jako jsou názvy míst a adresy, po složitější vědecké metody, které tvoří základ geodézie a průzkumu. Moderní metody georeferencování využívají pokročilé technologie, jako je GPS, což výrazně zvyšuje přesnost a efektivitu tohoto procesu.

Jednou z klíčových výzev v georeferencování je konverze mezi různými systémy georeferencování. Různé systémy mohou používat odlišné metody k určení polohy, což vyžaduje sofistikované techniky pro převod dat, aby byla zachována jejich přesnost a užitečnost. Správná konverze je nezbytná pro integraci a analýzu dat z různých zdrojů, což je základním stavebním kamenem pro efektivní využití geografických informačních systémů.

(Paul A. Longley, 2015)

**Princip:** Georeferencing je proces přiřazení skutečných světových souřadnic geografickým datům. Tento proces je zásadní v geografických informačních systémech (GIS), protože umožňuje přesné umístění dat na model Země.

Představte si, že máte mapu nebo letecký snímek oblasti, ale bez georeferencí - je to jen obrázek bez informace, kde přesně na Zemi se nachází. Georeferencování spočívá v identifikaci specifických bodů na tomto obrázku (například křižovatky, vrcholy hor) a

přiřazení těchto bodů k jejich odpovídajícím souřadnicím na Zemi, což může být například v systému GPS.

Jakmile jsou tyto body identifikovány a přiřazeny, mohou být použity k transformaci a zarovnání celého obrázku nebo mapy tak, aby odpovídal skutečnému umístění na Zemi. To umožňuje integrovat data z různých zdrojů a používat je pro přesnou prostorovou analýzu a rozhodování v GIS.

Tento proces je klíčový pro zajistění, že geografická data jsou relevantní a užitečná, protože bez správného georeferencování by data neměla kontext a bylo by je obtížné správně interpretovat a použít.

(Paul A. Longley, 2015)

## 1.4 Trendy a budoucí směrování

Trendy a budoucí směrování geografických dat se v roce 2024 zaměřují na několik klíčových aspektů, které jsou významné pro podnikání a technologický pokrok. Mezi hlavní trendy patří integrace technologií, strategická odolnost, inovativní růstové strategie a důraz na lidský přístup k technologiím. Tyto faktory jsou nezbytné pro navigaci v rychle se vyvíjejícím obchodním a technologickém prostředí. (Večeřa, 2024)

Jedním z klíčových trendů je ochrana investic do technologií s důrazem na dlouhodobou udržitelnost a bezpečnost. Dále se zdůrazňuje význam "Vzestupu stavitelů", což znamená odemykání kreativního potenciálu napříč organizacemi pomocí odvětvově specifických technologií a demokratizace tvůrčího procesu. Rovněž se klade důraz na sjednocení technologií s měnícími se požadavky zákazníků a trhů, což zdůrazňuje agilitu a schopnost rychle reagovat na změny. (Večeřa, 2024)

V oblasti geografických dat jsou významné trendy spojené s využitím velkých dat. Geografická data jsou stále komplexnější a obsahují více informací, což zvyšuje potřebu jejich efektivního indexování a analýzy. Například analýza finančních transakcí může pomoci odhalit podvody, zatímco agregace dat z meteorologických stanic nebo bleskových úderů umožňuje lepší vizualizaci a porozumění těmto jevům. (Souček, 2021)

Kvantitativní analýza geografických dat se stává stále důležitější. Současné trendy v této oblasti zahrnují rozvoj nových metod a přístupů, jako je prostorová analýza a časoprostorové kostky, které umožňují sledovat vývoj veličin jak v prostoru, tak v čase. (Netrdová, 2010)

## 1.5 BIG DATA a geografická data

V současné době se oblast zpracování velkých dat rychle rozvíjí, čímž se otevírají nové možnosti pro analýzu a zpracování geoprostorových dat. Přechod od tradičních Big Data 1.0 systémů, jako je Hadoop, k modernějším a více specializovaným Big Data 2.0 systémům, umožňuje efektivnější zpracování dat v reálném čase a s vysokou variabilitou typů dat, což je klíčové pro geoprostorová data. (Sakr, 2020)

Sakr ve své knize zdůrazňuje, že "i přes vysoká očekávání vůči slibům a potenciálům paradigmatu velkých dat, stále existuje mnoho výzev na cestě k plnému využití jejich síly. Typické charakteristiky velkých dat, jako jsou rozmanité typy mající složité vzájemné vztahy a ne nutně konzistentní vysokou kvalitu dat, vedou k významnému nárůstu výpočetní složitosti a požadovaného výpočetního výkonu. Proto se tradiční úlohy zpracování a analýzy dat, včetně vyhledávání, analýzy sentimentu a sémantické analýzy, stávají stále složitějšími ve srovnání s tradičními daty." (Sakr, 2020)(překlad autora) Tento pohled podtrhuje komplexitu, s kterou se setkáváme při práci s geoprostorovými daty v naší diplomové práci, a zdůrazňuje potřebu sofistikovanějších metod a technologií pro jejich zpracování a analýzu.

# 2 OpenStreetMap

OpenStreetMap je inovativní projekt, který se snaží poskytnout uživatelům po celém světě přístup k detailním a aktualizovaným mapám. Na rozdíl od tradičních mapových služeb, které mohou obsahovat omezení použití nebo mohou být zastaralé, OSM umožňuje komunitě uživatelů přispívat a aktualizovat mapová data v reálném čase. Tím se stává jedinečným zdrojem pro řidiče, turisty, vědce a firmy potřebující spolehlivé geografické informace.

Projekt se spoléhá na dobrovolníky, kteří mapují různé oblasti světa, od velkých měst po odlehlé regiony, což zajišťuje, že mapy jsou co nejvíce komplexní a aktuální. Data jsou poté volně dostupná pro každého, kdo je potřebuje, což umožňuje vznik nových aplikací a služeb, které mohou využívat tato data.

OpenStreetMap také podporuje vzdělávací a humanitární projekty poskytováním map a dat, které jsou klíčové pro plánování a realizaci projektů po celém světě. Projekt si klade za cíl být nejen zdrojem map, ale také platformou pro sdílení znalostí a podporu globální komunity zájemců o mapování a geografii. (openstreetmap.cz, 2015)

## 2.1 Historie

Steve Coast dal v roce 2004 vzniknout projektu OpenStreetMap, který se zpočátku soustředil na kartografii Velké Británie. V této zemi a dalších státech sice existovaly rozsáhlé datové soubory vytvořené státními institucemi financovanými z veřejných prostředků, jako například Ordnance Survey, avšak tyto informace nebyly široce a bezplatně dostupné. Proto byla 22. srpna 2006 zřízena Nadace OpenStreetMap, jejímž cílem bylo podporovat šíření, rozvoj a volné užívání geoprostorových informací, aby je mohl využívat a sdílet kdokoliv.

Při úpravách map v rámci OpenStreetMap se dobrovolníci spoléhali na java-aplet na webové stránce projektu nebo na samostatné programy, které umožňovaly práci bez připojení k internetu a využívání vlastních terénních průzkumů, záznamů z GPS a snímků ze satelitů, jež jsou ve veřejné doméně. Mezi tyto offline editory patřil zejména JOSM, jenž byl průběžně zdokonalován a zůstává jedním z hlavních nástrojů pro editaci. V prosinci 2006 pak společnost Yahoo dala OpenStreetMap k dispozici své letecké snímky pro podklad při tvorbě map. Následně, o šest měsíců později, byl na webových stránkách projektu uveden nový online editor Potlatch, který měl za úkol usnadnit zapojení nových přispěvatelů do projektu.

Metody pro import a export dat se postupně rozrůstaly, a tak projekt do roku 2008 vyvinul nástroje umožňující přenášet data z OpenStreetMap do přenosných navigačních zařízení, čímž byly nahrazeny starší a proprietární mapové podklady.

V listopadu 2010 byla představena nová verze editoru Potlatch 2, a s možností využívání vertikálních leteckých snímků od Microsoft Bing se otevřely další možnosti pro tvorbu map. V květnu 2013 byl zaveden nový online editor iD, který se vyznačuje jednoduchostí a uživatelskou přívětivostí. (OpenStreetMap Wiki, 2023)

### 2.1.1 Historie OSM v České republice

První pokusy o vytváření mapových dat v České republice se datují do roku 2005. Skutečný rozvoj mapování nastal po importu hlavních a vedlejších silnic, který umožnila firma HELP SERVICE - REMOTE SENSING. K dalším klíčovým importům patřil přenos dat o lesích (UHUL), vodních plochách a tocích (DIBAVOD), adresách (RÚIAN) a zemědělských plochách (LPIS).

Mapování také zahrnovalo využití gpx záznamů, leteckých snímků a dalších zdrojů, díky čemuž byly do OpenStreetMap začleněny informace o železniční síti, energetických vedeních, turistických a cyklistických trasách. (OpenStreetMap Wiki, 2020 )

**Významné mezníky:**

- 2005: Počátek mapování v ČR, první experimenty.
- 16.10.2007: Přidány hlavní a vedlejší silnice.
- 25.10.2007: Přidány obce.
- 2008: Ruční doplnění vedlejších silnic z leteckých snímků a katastrálních map.
- 08/2008: Přidány lesní plochy - UHUL.
- Od roku 2009: Částečně automatizovaný import budov pomocí nástroje Tracer.
- 02/2010: Přidány administrativní hranice.
- 2010-2011: Přidány vodní plochy a toky - DIBAVOD.
- 2010-2011: Částečně automatizovaný import adres.
- 18.10.2013: Začátek překladu dokumentace do češtiny.
- 2014: Přidány adresní body z RÚIAN.
- 2014-2015: Částečně automatizovaný import zemědělských ploch z LPIS.
- 2016: Spuštěn nový web openstreetmap.cz a zavedení Taskmanu pro kontrolu turistických tras.

- 21.05.2016: Konání prvního ročníku konference State of the Map CZ+SK.
- 7.9.2016: Pořádání prvního setkání s pivem v Brně a Praze. (OpenStreetMap Wiki, 2020 )

## 2.2 Principy OpenStreetMap

**Otevřenost**: OSM je založen na ideji otevřených dat, což znamená, že všechna mapová data jsou volně dostupná pro použití, úpravu a sdílení jakýmkoli způsobem, za předpokladu, že uživatelé citují zdroj a sdílejí odvozená díla za stejných podmínek. Tato otevřenost podporuje inovace a umožňuje širokou škálu použití od komerčních aplikací po neziskové projekty.

**Spolupráce**: OSM je komunitní projekt, kde data vytvářejí, kontrolují a aktualizují dobrovolníci z celého světa. Tato spolupráce zahrnuje jak jednotlivce, tak organizace. Každý může přispět, ať už přímým mapováním, úpravou stávajících dat, nebo pomocí v mapování.

**Přístupnost**: OSM je navržen tak, aby byl přístupný široké veřejnosti. Webová rozhraní a editory, jako jsou iD nebo JOSM, usnadňují nováčkům zapojení do projektu. Vzdělávací materiály a komunitní podpora jsou k dispozici pro všechny úrovně dovedností.

**Udržitelnost**: OSM je navržen s ohledem na dlouhodobou udržitelnost. Nadace OpenStreetMap a místní komunity po celém světě usilují o udržení a rozvoj projektu prostřednictvím organizace událostí, získávání finančních prostředků a zajišťování infrastruktury.

**Flexibilita**: OSM umožňuje reprezentaci široké škály geoprostorových dat, od silnic a budov až po přírodní prvky a bodová zařízení. Uživatelé mohou přidávat a upravovat různé typy dat podle potřeby, což činí OSM extrémně adaptabilním na různé mapovací projekty.

**Transparentnost**: Veškeré změny v mapových datech jsou zaznamenány a jsou veřejně dostupné, což umožňuje sledování historie a případné korekce. Tento přístup podporuje spolehlivost a důvěru v data.

**Nezávislost**: Ačkoli OSM spolupracuje s různými organizacemi a přijímá data od třetích stran, projekt si zachovává nezávislost a zaměřuje se na poskytování nezkreslených a nekomerčních mapových dat.

## 2.3 Ukládání dat

V OpenStreetMap (OSM) jsou data uložena ve vlastním XML formátu, který využívá globální souřadnicový systém WGS84. Tento formát umožňuje zachovat jak geometrické tak i atributové informace prvků. Atributy jsou zaznamenány v tagovacích značkách <tag>, které

poskytují podrobné informace o mapových prvcích. Datový model OSM, který slouží k reprezentaci fyzického světa, se skládá z elementárních stavebních bloků nazývaných prvky <elements>. Mezi tyto základní prvky patří uzly <node>, které reprezentují konkrétní body na mapě, cesty <way>, které spojují tyto body do liniových nebo plošných struktur, a relace <relation>, jež umožňují vytvářet vazby mezi různými prvky. Data z OSM je možné exportovat a konvertovat do různých vektorových formátů, jako jsou SHP nebo PBF, což umožňuje jejich široké využití.

Data OpenStreetMap (OSM) se shromažďují z mnoha zdrojů a jsou dostupná v různých formátech, včetně týdenních aktualizací velké OSM databáze a menších souborů vhodných pro stahování. Mezi důležité nástroje pro práci s OSM daty patří Osmosis, který umožňuje zpracovávat a analyzovat data OSM. Historie editací v OSM je komplexní, ale ne každá změna se okamžitě odráží v nejnovější verzi datasetu. Po uvedení OSM API verze 0.6 už nejsou do historického záznamu OSM zahrnuty příspěvky, které nebyly schváleny pod Open Database License (ODbL). (Barron, a další, 2014)

### 2.3.1 Uzly (Nodes)

Uzel (v anglickém originále "Node") je v datovém modelu OpenStreetMap základním prvkem, který reprezentuje specifický bod na zemském povrchu. Každý uzel je definován svými geografickými souřadnicemi, tedy zeměpisnou šířkou a délkou, v souřadnicovém systému WGS 84. Uzly mohou být použity samostatně k označení konkrétních bodových zájmových míst, jako jsou například poštovní schránky, lavičky nebo stromy. V kontextu větších struktur uzly slouží jako koncové nebo spojovací body cest, které mohou reprezentovat různé liniové prvky, jako jsou silnice, řeky nebo železniční tratě, nebo mohou tvořit hranice plošných prvků, jako jsou budovy, jezera nebo parky.

Uzly mohou nést různé tagy (značky), které poskytují dodatečné informace o daném bodu, jako jsou název, typ objektu, materiál, barva a mnoho dalších atributů. Tagy tedy umožňují přidat k uzlu podrobné sémantické informace a kategorizovat jej do různých typů geografických nebo umělých prvků.

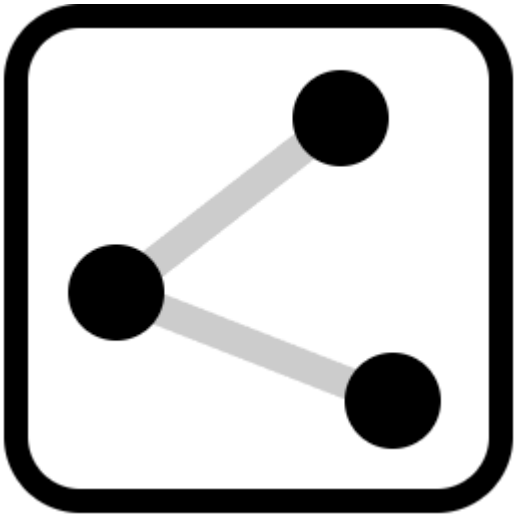
Každý uzel v OpenStreetMap má jedinečné identifikační číslo (ID), které umožňuje jeho jednoznačnou identifikaci v rámci celé databáze. Toto ID je klíčové pro propojování uzlů s dalšími prvky datového modelu OSM, jako jsou cesty a relace, a pro udržení integrity a soudržnosti mapových dat. (OpenStreetMap Wiki, 2020)

```
<node id="25496583" lat="51.5173639" lon="-0.140043" version="1" changeset="203496" user="80n" uid="1238"
visible="true" timestamp="2007-01-28T11:40:26Z">
    <tag k="highway" v="traffic_signals"/>
</node>
```

*Obrázek 1: Příklad uzlu (Node) v OSM Zdroj: (OpenStreetMap Wiki, 2020)*

### 2.3.2 Cesty (Ways)

Cesta v kontextu OpenStreetMap (OSM) je fundamentální prvek, který slouží k vyjádření různorodých geografických objektů na mapě. Tento prvek je tvořen spojením tří nebo více bodů, známých jako uzly, do jedné sekvence, čímž vzniká linie nebo v případě uzavření sekvence uzlů i plošný útvar.



*Obrázek 2: OSM element way (cesta). Zdroj: (OpenStreetMap Wiki, 2024 )*

Cesty jsou využívány k mapování liniových objektů jako jsou silnice, pěší stezky, vodní toky či železniční tratě, ale také k definování obrysů plošných prvků, jako jsou budovy, jezera nebo parky, kde uzavřená sekvence uzlů tvoří hranici daného objektu.

Každá cesta v OSM je jednoznačně identifikována svým unikátním identifikačním číslem (ID), což umožňuje její odlišení od ostatních prvků v databázi. Aby bylo možné přesně specifikovat charakteristiku a funkci cesty, přiřazují se jí tagy - páry klíč: hodnota, které poskytují detailní informace o typu cesty, jejím povrchu, názvu a dalších atributech. Tyto tagy jsou zásadní pro správnou interpretaci cest a jejich adekvátní zobrazení na mapách.

Cesty mohou také hrát klíčovou roli v rámci složitějších struktur, známých jako relace, kde můžou zastávat specifické funkce nebo role, například jako části komplexních tras nebo jako součásti definic hranic oblastí. (OpenStreetMap Wiki, 2024 )

### 2.3.3 Relace (Relation)

Relace (anglicky "Relation") v OpenStreetMap (OSM) je pokročilý a flexibilní datový prvek, který umožňuje definovat vztahy mezi různými uzly (nodes), cestami (ways) a dokonce i jinými relacemi. Tento prvek je klíčový pro vyjádření složitějších struktur a konceptů, které nelze adekvátně popsat pouze pomocí uzlů a cest. Relace se skládá z jednoho nebo více členů (members), přičemž každý člen má přiřazenou roli (role), která specifikuje jeho funkci v rámci relace. Členy relace mohou být uzly, cesty nebo jiné relace, což umožňuje vytvářet velmi komplexní a vícevrstvé struktury.

#### Typy Relací

Relace mohou reprezentovat širokou škálu geografických a mapových konceptů, včetně, ale nikoli omezeně na:

**Multipolygonové objekty:** Pro složité plošné objekty, které se skládají z více cest, jako jsou například parky s jezery uvnitř, kde jezera tvoří "díry" v ploše parku.

**Trasy veřejné dopravy:** Kde relace definuje trasu autobusu, tramvaje nebo vlaku, spojující jednotlivé zastávky (uzly) pomocí určených cest.

**Soustavy cest:** Například souvislé silniční nebo turistické trasy, kde relace spojuje segmenty cest do jednoho logického celku.

#### Identifikace a Struktura

Každá relace je unikátně identifikována svým ID a její struktura umožňuje flexibilní a dynamické modelování mapových dat. Struktura relace umožňuje mapovat reálný svět s větší přesností a detailností, reflektujíc závislosti a vztahy mezi různými mapovými prvky.

Relace v OSM představují mocný nástroj pro pokročilé mapování, umožňující uživatelům efektivně reprezentovat komplexní struktury a systémy, které jsou v reálném světě často vzájemně propojené a závislé. (OpenStreetMap Wiki, 2023 )

### 2.3.4 Tagy

Tagy v OpenStreetMap (OSM) jsou základním mechanismem pro přiřazení sémantických informací k uzlům, cestám a relacím, čímž se určují vlastnosti a charakteristiky mapových prvků. Každý tag se skládá z páru klíč:hodnota, kde klíč určuje typ informace a hodnota specifikuje tu informaci.

**Klíče a Hodnoty**

- Klíč (Key): Identifikuje kategorii informace, kterou tag poskytuje, například `highway` pro typy silnic, `amenity` pro veřejné vybavení nebo `building` pro budovy.

- Hodnota (Value): Konkrétně specifikuje vlastnost dané kategorie, jako je `residential` pro rezidenční silnice v rámci klíče `highway`, nebo `school` pro školu v rámci klíče `amenity`.

**Použití Tagů**

Tagy umožňují mapování širokého spektra informací od základních typů objektů, jako jsou silnice, řeky nebo parky, až po detailní charakteristiky, jako jsou materiály, barvy, názvy a další specifické atributy. Použití tagů je velmi flexibilní a komunitou řízené, což znamená, že se mohou vytvářet nové klíče a hodnoty podle potřeby.

**Tagovací Schémata**

V OSM existují doporučená tagovací schémata pro běžné typy objektů a situace, ale uživatelé mohou přidávat vlastní tagy, pokud existující schémata nevyhovují jejich potřebám. Schémata pomáhají udržet konzistenci dat a usnadňují jejich použití a analýzu.

**Význam Tagů**

Tagy jsou zásadní pro užitečnost a bohatost dat v OSM, protože poskytují nejen základní informace o typu objektu, ale také umožňují detailní popis jeho vlastností a kontextu. Efektivní tagování je klíčové pro přesné a informativní mapování, umožňující vytvářet bohaté a komplexní geoprostorové informace přístupné pro různé aplikace a služby. (OpenStreetMap Wiki, 2023)

## 2.4 Komunita a spolupráce v OpenStreetMap

OpenStreetMap (OSM) je komunitní projekt, který se zaměřuje na vytváření svobodné a editovatelné mapy světa. Uživatelé z celého světa přispívají svými lokálními znalostmi, sběrem dat a mapováním oblastí, což činí OSM dynamickým a neustále se rozvíjejícím projektem. Spolupráce v rámci komunity OSM zahrnuje různé aktivity, od individuálního mapování pořádaného jednotlivými uživateli až po organizované mapovací akce, jako jsou mapovací párty, které propojují místní komunity a podporují sběr dat v konkrétních oblastech.

Uživatelé OSM využívají různé nástroje a technologie pro přidávání a aktualizaci dat. Mezi tyto nástroje patří webové editory, jako je iD editor, který umožňuje snadné mapování přímo v prohlížeči, a pokročilejší desktopové aplikace, jako je JOSM, které poskytují

rozsáhlejší nástroje pro editaci. Kromě toho komunita využívá GPS zařízení pro přesné mapování terénu a fotografování, které umožňuje přidávání detailních informací do mapy. (OpenStreetMap Wiki, 2024)

„Největší síla OSM bude vždy spočívat v obrovském počtu přispěvatelů. Tisíce těchto přispěvatelů shromáždily a vytvořily jedny z nejlepších uličních a topografických dat na světě, a to bez nákladných týmů profesionálních geodetů nebo špičkového vybavení. Vzhledem k tomu, že svět a městské i přírodní prostředí se každý den mění, mají přispěvatelé OSM možnost zobrazit tento měnící se svět v mapě a databázi, které jim patří. mapovací párty.“ (Foody, a další, 2017)(překlad autora)

Kolaborace uživatelů v OSM představuje klíčovou složku projektu, která odlišuje OSM od jiných online platforem pro tvorbu obsahu, jako je Wikipedia. Na rozdíl od Wikipedie, kde jednotlivci vytvářejí většinu obsahu na různých místech, komunita OSM organizuje řadu lokálních dílen, známých jako "mapovací párty". Tyto události jsou zaměřeny na vytváření a anotaci obsahu pro lokalizované geografické oblasti a jsou navrženy tak, aby nové uživatele a přispěvatele seznámily s komunitou prostřednictvím praktických zkušeností se sběrem, zpracováním a nahráváním dat do projektu OSM. (Haklay, a další, 2008)

Mapovací párty mohou nabývat podoby neformálních a malých setkání, která trvají několik hodin a zaměřují se na dokončení chybějících prvků v malé definované čtvrti, až po ambicióznější snahy, které trvají několik dní a zahrnují desítky účastníků. Jedna z prvních mapovacích párty se konala na ostrově Isle of Wight u jižního pobřeží Anglie v květnu 2006, kde více než 30 účastníků z Evropy strávilo dva dny jízdou, cyklistikou a procházkami po ostrově s GPS přijímači, aby shromáždili úplné pokrytí silnic a pěších cest. Po shromáždění individuálních příspěvků, jejich zpracování a nahrání dat se objevila prakticky kompletní mapa ostrova. (Haklay, a další, 2008)

## 2.5 Vykreslování dat

Vizualizace dat je klíčovým prvkem v oblasti geoprostorových informací, umožňující efektivní interpretaci a sdílení složitých datových sad. V kontextu OpenStreetMap (OSM) poskytuje vizualizace neocenitelné nástroje pro zobrazení rozsáhlého množství informací shromážděných komunitou. Aplikace jako OpenTopoMap, OpenCycleMap, a OpenSnowMap demonstrují, jak lze data OSM vizualizovat pro specifické účely, jako je topografie, cyklistika, nebo zimní sporty, což uživatelům umožňuje přizpůsobit mapy svým konkrétním potřebám.

Díky flexibilitě a široké dostupnosti dat OSM, spolu s rozmanitostí nástrojů pro vizualizaci, se OSM stalo populárním zdrojem pro vytváření přizpůsobených mapových řešení. "Z anekdotických důkazů je zřejmé, že vizualizace dat OSM je jednou z nejpopulárnějších aplikací dat OSM," (Foody, a další, 2017 str. 45)(překlad autora) což poukazuje na široké využití a přijetí OSM v různých aplikacích. Vizualizace OSM dat umožňuje nejen zobrazit geografické informace, ale také přispět k rozvoji lokálních komunit a podporovat veřejné služby, jako je plánování městské infrastruktury a řízení katastrof.

Z těchto důvodů je vizualizace dat OSM neocenitelným nástrojem pro geoprostorové analýzy, plánování a komunitní rozvoj. Flexibilita a rozmanitost vizualizačních nástrojů spolu s aktivním zapojením komunity přispívají k neustálému zlepšování a aktualizaci dat, což z OSM činí dynamickou a životaschopnou platformu pro geoprostorové informace. (Foody, a další, 2017)

### 2.5.1 Renderování

Renderování v kontextu OpenStreetMap je proces, při kterém se surová geoprostorová data transformují na vizuální mapy. Tento proces umožňuje vytvářet mapy různých stylů a účelů, od standardních silničních map po specializované mapy, jako jsou cyklistické nebo turistické mapy. Flexibilita v renderovacích nástrojích a přístupu umožňuje komunitě OpenStreetMap přizpůsobovat výstupy specifickým potřebám a preferencím, což podporuje širokou škálu aplikací od navigace po urbanistické plánování. Významnou roli v tomto procesu hrají i open-source nástroje a knihovny, které jsou komunitě volně k dispozici pro vývoj a inovace. (OpenStreetMap Wiki, 2024 )

Proces renderování v OpenStreetMap zahrnuje několik kroků, kde klíčovým je převod datových bodů, linií a polygonů na grafické reprezentace na mapě. Tento proces se řídí předem definovanými pravidly a stylemi, které určují, jak budou různé typy dat (jako jsou silnice, budovy, vodní plochy) vizualizovány. Výsledná mapa je pak kompilována a zobrazována uživatelům, přičemž se mohou používat různé renderovací nástroje a softwary, od serverových řešení po klienty na straně uživatele. (OpenStreetMap Wiki, 2024 )

Na rozdíl od tradičních mapovacích služeb, OpenStreetMap umožňuje uživatelům přispívat a upravovat mapová data, což má za následek dynamicky se rozvíjející databázi geoprostorových informací. Tento otevřený přístup vyžaduje efektivní renderovací systémy, které dokážou aktualizovat mapy v reálném čase a zobrazovat přesné a aktuální informace. Renderovací proces v OpenStreetMap tak představuje klíčovou složku v ekosystému sdílení

geoprostorových dat, umožňující komunitě neustále zlepšovat a aktualizovat mapový obsah. (OpenStreetMap Wiki, 2024 )

Serverové a příkazové řádkové nástroje:

**Carto:** Převádí CartoCSS styly do XML stylů Mapniku, používá JavaScript.

**Mapnik:** Velmi oblíbený serverový 2D mapový vykreslovač pro Windows, macOS a Linux, podporuje mnoho programovacích jazyků a je určen pro rychlou generaci dlaždic na výkonných serverech.

**MapOSMatic:** Nástroj, který může vykreslit mapy s mřížkou a rejstříkem ulic, běží na různých platformách a je napsán v Pythonu s využitím Django a Mapniku.

**Mrender:** Pravidly řízený vykreslovací engine napsaný v C.

**TileServer GL:** Server pro rasterové a vektorové dlaždice, funguje na různých platformách.

**TileSweep:** Tile server s předběžným vykreslováním, používá libmapnik a běží na Linuxu a macOS.

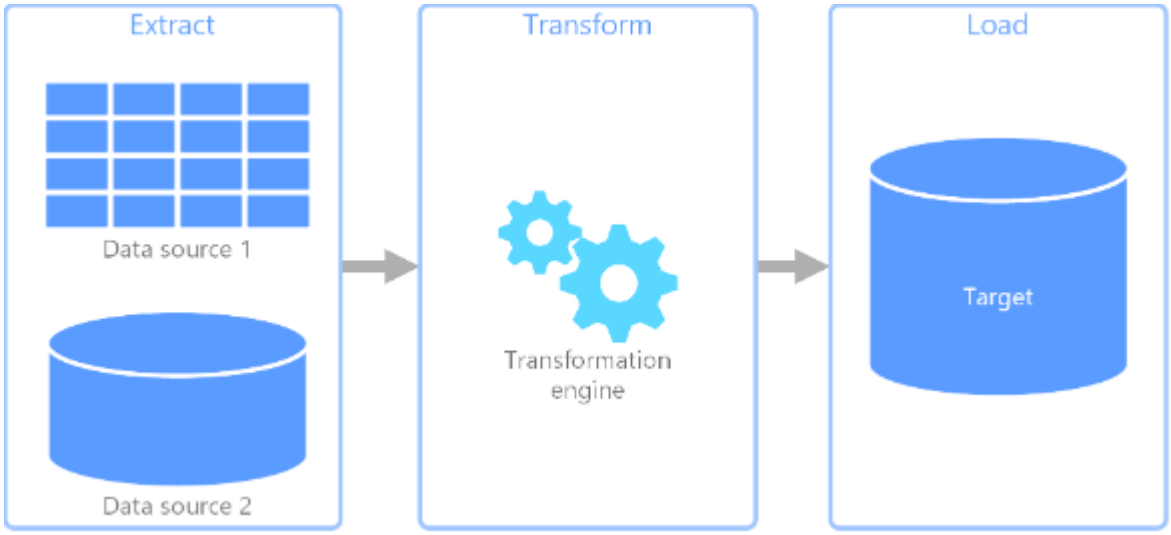
**Map Machine:** Mapový vykreslovač pro OpenStreetMap s vlastními ikonami, zaměřený na zobrazení co nejvíce tagů, funguje na různých platformách.

Každý nástroj je určen pro konkrétní platformy a má specifické využití v kontextu práce s mapovými daty OpenStreetMap. (OpenStreetMap Wiki, 2024 )

# 3 ETL proces

ETL je proces používaný v oblasti zpracování a analýzy dat, který zahrnuje tři základní kroky: extrakci dat z různých zdrojů, jejich transformaci do požadovaného formátu a nakonec načítání transformovaných dat do cílového úložiště nebo systému. Tento proces umožňuje organizacím efektivně spravovat a analyzovat data, která mohou pocházet z různorodých a rozptýlených zdrojů, a integrovat je do jednotného, konzistentního formátu vhodného pro analýzu a rozhodování.

V obecném smyslu ETL představuje způsob, jakým firmy přistupují k datům, jež potřebují pro své operace, analýzy a rozhodovací procesy. Procesy ETL se staly základem moderních systémů Business Intelligence (BI), datových skladů a dalších analytických aplikací, které vyžadují komplexní manipulaci s daty. ETL umožňuje organizacím transformovat neuspořádaná a rozptýlená data v užitečné informace, které podporují strategické plánování a operativní efektivitu.



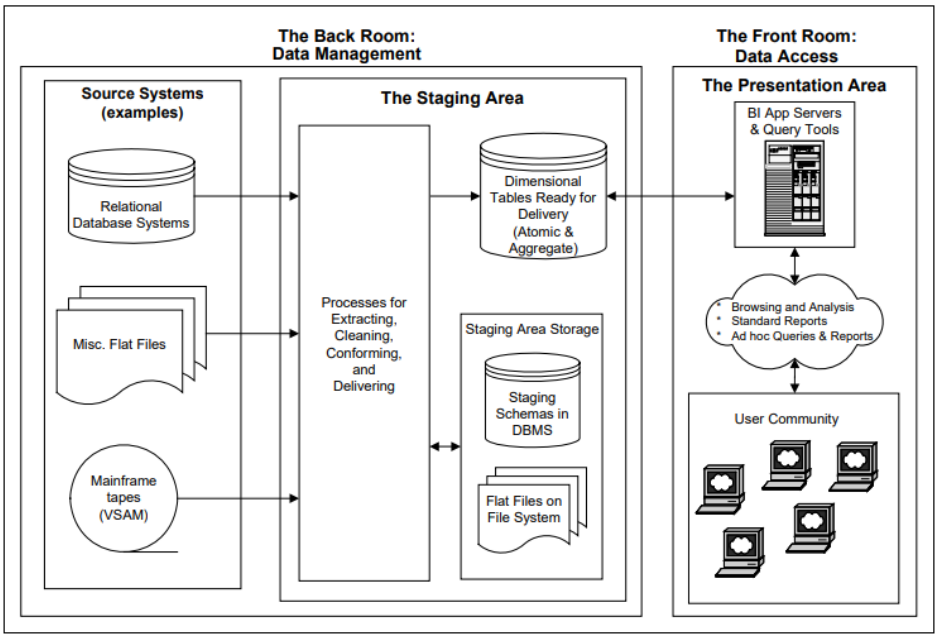

*Obrázek 3: ETL proces. Zdroj: (Microsoft)*

## 3.1 Datové sklady

Datové sklady jsou centrálními repozitáři informací, které se využívají k analýze dat za účelem informovanějšího rozhodování. Data do datového skladu přitékají z transakčních systémů, relačních databází a dalších zdrojů, obvykle v pravidelných intervalech. Analytici, inženýři dat, vědci v oblasti dat a rozhodovatelé přistupují k datům prostřednictvím nástrojů pro business intelligence (BI), SQL klientů a dalších analytických aplikací. Datové sklady usnadňují tyto reporty, dashboardy a analytické nástroje tím, že efektivně ukládají data tak, aby

minimalizovaly vstup a výstup dat a rychle doručovaly výsledky dotazů stovkám a tisícům uživatelů současně. (Aws Amazon, 2024)

Architektura datového skladu závisí na konkrétních potřebách organizace a obvykle zahrnuje několik vrstev, včetně klientského rozhraní pro prezentaci výsledků, analytického enginu pro přístup a analýzu dat a databázového serveru, kde jsou data ukládána a načítána. Data mohou být ukládána různými způsoby, například často přistupovaná data jsou uložena na rychlých úložištích, jako jsou SSD disky, zatímco méně často přistupovaná data jsou uložena v levnějším objektovém úložišti, jako je Amazon S3. (Aws Amazon, 2024)




*Obrázek 4: Zadní a přední místnost datového skladu. Zdroj: (Kimball, a další, 2004)*

## 3.2 Extrakce

Extrakce dat je klíčovým prvním krokem v procesu ETL (Extract, Transform, Load), kde se data získávají z různých zdrojů. Tento krok je zásadní pro další zpracování a analýzu dat, jelikož kvalita a přesnost extrahovaných dat přímo ovlivňují celkovou účinnost a výkonnost procesů transformace a nahrávání. Pro efektivní extrakci je důležité dobře porozumět zdrojovým datům, včetně typů dat a schémat, a zvolit správné metody a nástroje pro jejich extrakci, aby byla zajištěna jejich integrita a relevantnost.

Extrakce dat je kritickým procesem v ETL cyklu, který vyžaduje důkladné plánování a přípravu. Klíčem k úspěchu je vytvoření logického datového mapování před fyzickou implementací. Logické mapování definuje vztahy mezi zdrojovými poli a cílovými poli v datovém skladu, odhaluje jaká data a transformace jsou potřebná pro naplnění požadavků

datového skladu. Kromě toho, je důležité identifikovat a dokumentovat zdrojové systémy, analyzovat je pomocí nástrojů pro profilování dat a ověřit jejich kvalitu a připravit je na transformační procesy. Dále je potřeba validovat výpočty a vzorce s koncovými uživateli a zabezpečit, že data jsou správně interpretována a používána v souladu s obchodními pravidly. (Kimball, a další, 2004 str. 58)

Vytváření logické datové mapy a její důsledné sledování zajišťuje, že všechny zdroje dat jsou správně pochopeny a dokumentovány, což je základ pro efektivní extrakci dat. Tento proces pomáhá předcházet problémům v budoucích fázích ETL a zajišťuje, že data jsou přesně a účelně extrahována pro potřeby datového skladu. (Kimball, a další, 2004 str. 59)

## 3.3 Transformace

Očišťování a transformace dat jsou kritickými kroky v procesu ETL, které přímo ovlivňují kvalitu a použitelnost dat ve finálním datovém skladu. Proces očišťování dat zahrnuje identifikaci, korekci nebo odstranění chybných, neúplných nebo irelevantních dat, která mohou negativně ovlivnit analýzu dat. To vyžaduje pečlivou analýzu datových zdrojů, použití sofistikovaných nástrojů pro profilování dat a vývoj robustních pravidel pro validaci dat. Je důležité, aby tým odpovědný za ETL procesy rozuměl obchodním pravidlům a datovým standardům, aby mohl účinně identifikovat a řešit problémy s datovou kvalitou. (Kimball, a další, 2004)

Transformace dat je proces převodu dat do požadovaného formátu nebo struktury pro následné použití nebo analýzu. To může zahrnovat změny datových typů, agregaci, normalizaci a další metody přepracování dat tak, aby lépe odpovídala požadavkům koncového systému, jako je datový sklad. Transformace je často prováděna pomocí komplexních ETL nástrojů, které umožňují automatické i manuální zpracování datových sad. (Kimball, a další, 2004)

V obou těchto fázích je klíčová komunikace a spolupráce mezi různými týmy v rámci organizace, včetně datových analytiků, databázových administrátorů a obchodních uživatelů. Společným cílem by mělo být dosažení vysoké úrovně datové integrity, což umožňuje organizaci důvěřovat datům a používat je k podložení důležitých obchodních rozhodnutí. (Kimball, a další, 2004)

## 3.4 Zápis

Proces načítání dat do datového skladu představuje značnou výzvu, obzvláště při prvotním načtení velkého objemu dat do nové tabulky. Jedním ze základních kroků je oddělení

vkládaných záznamů (inserts) od aktualizací stávajících dat (updates). Mnohé ETL nástroje a databázové systémy nabízejí funkci „update else insert“, která je sice pohodlná a zjednodušuje logiku toku dat, ale je známa svou pomalostí. Proto by ETL procesy vyžadující aktualizace existujících dat měly obsahovat logiku, která tyto záznamy oddělí, aby mohl být následný proces načítání co nejefektivnější. (Kimball, a další, 2004)

## 3.5 ETL v kontextu geografických dat

Inženýrství geoprostorových dat v GIS se zabývá správou, zpracováním, čištěním a analýzou geoprostorových dat, což je oblast velmi blízká geoprostorové datové vědě. Inženýři dat se soustředí více na implementaci procesů inženýrství dat, zatímco datoví vědci se zaměřují na objevování a průzkum dat. Proces v GIS zahrnuje extrakci a kompilaci dat z více zdrojů, jejich transformaci do užitečného formátu pro podnikání a nahrání do datového skladu. Tato praktická a detailně orientovaná profese vyžaduje od inženýrů trpělivost a schopnost řešit problémy s důrazem na pečlivost, přičemž přidání geoprostorové složky zvyšuje složitost prostorové analytiky v cloudu. Dnes jsme se pouze lehce dotkli potenciálu inženýrství dat v GIS. (GIS Geography, 2023)

Je důležité zmínit, že integrace a zpracování dat geografických je komplikovanější než práce s daty negeografickými. Je třeba, aby data byla přesně topologicky zarovnána a respektovala prostorové integrity. Významná je správná transformace prostorových referenčních systémů a metod, přesnost geometrie objektů a konzistence map v různých měřítcích. Odborné znalosti v oblasti prostorového referencování jsou nezbytné, proces nelze automatizovat plně. Uživatelé často potřebují čistit a integrovat prostorová data z různých historických období, což vyžaduje úvahy o kompromisech, protože se mohou měnit základní prostorové jednotky a historická data nejsou vždy dostupná. Vývoj technologií ETL pro prostorová data nabývá na důležitosti a může zahrnovat nástroje pro lepší zpracování a agregaci prostorových měření. (Miller, a další, 2009)

Přestože jsou tu výzvy, je stále možné vyvíjet jednoduché aplikace pro prostorové databáze, pokud se udrží kartografické požadavky v rozumných mezích. Mnoho aplikací již úspěšně minimalizovalo výše uvedené problémy, zejména při práci s pečlivě regulovanými administrativními daty. Na druhou stranu, databáze přírodních jevů nebo databáze bez historických dat vyžadují speciální přístupy, jako jsou například časově omezené databáze nebo databáze s různorodou kvalitou dat. Většina úsilí při budování prostorových databází se

soustředí na prostorové ETL procesy a kvalita stávajících dat má významný dopad na návrh a konstrukci těchto databází. (Miller, a další, 2009)

### 3.5.1 Známé nástroje pro správu geografických dat

Existuje mnoho ETL nástrojů, které jsou speciálně navrženy pro práci s geografickými daty, nebo které mají rozšíření či moduly podporující geoprostorové funkce. Zde jsou některé z nejpoužívanějších nástrojů:

**FME (Feature Manipulation Engine) od SAFE Software**

FME je silný a flexibilní ETL nástroj speciálně navržený pro manipulaci a transformaci geoprostorových dat. Podporuje stovky formátů (včetně ESRI Shapefile, GeoJSON, KML a mnoha dalších) a umožňuje komplexní transformace dat, včetně reprojekce, filtrace, agregace a dalších geoprostorových operací.

**Talend Open Studio**

Talend je open-source ETL nástroj, který nabízí silnou podporu pro integraci dat, včetně geoprostorových dat. Umožňuje transformaci a integraci dat z různých zdrojů a může být rozšířen o geoprostorové komponenty pro specifické úkoly.

**GeoKettle**

GeoKettle je geoprostorová odnož populárního open-source ETL nástroje Pentaho Data Integration (Kettle). Poskytuje rozsáhlou podporu pro geoprostorová data, umožňuje extrakci, transformaci a načítání geoprostorových dat v různých formátech a podporuje prostorové operace jako jsou prostorové filtry, transformace a agregace.

**ArcGIS Data Interoperability**

Tento rozšířený modul pro ArcGIS od Esri umožňuje snadnou integraci, migraci a transformaci geoprostorových dat. ArcGIS Data Interoperability poskytuje nástroje pro vizuální modelování ETL procesů a podporuje širokou škálu formátů geoprostorových dat.

**PostGIS**

PostGIS je rozšíření open-source objektově-relační databáze PostgreSQL, které přidává podporu pro geoprostorová data. Ačkoli primárně slouží jako databázové úložiště, lze jej použít v kombinaci s dalšími nástroji pro ETL procesy, které zahrnují geoprostorová data.

### QGIS

QGIS je open-source geografický informační systém, který může být použit pro některé ETL úkoly s geoprostorovými daty díky svým rozsáhlým možnostem importu, exportu a transformace dat. QGIS podporuje širokou škálu vektorových, rastrů a databázových formátů.

### GDAL/OGR

GDAL (Geospatial Data Abstraction Library) a OGR (OGR Simple Features Library) jsou knihovny poskytující nástroje pro čtení, zápis a transformaci rastrů a vektorových geoprostorových datových formátů. Tyto nástroje lze použít samostatně nebo ve skriptech pro automatizaci ETL procesů.

### Osm2pgsql

Osm2pgsql je robustní a výkonný nástroj používaný pro konverzi dat OpenStreetMap (OSM) do PostGISu, což je prostorové rozšíření PostgreSQL. Tento nástroj je nezbytný pro import velkých množství OSM dat a jejich efektivní uložení v relační databázi. Díky své schopnosti pracovat s velkými datovými sadami a flexibilitě v mapování OSM dat na uživatelsky definované schéma je osm2pgsql široce používán v GIS aplikacích po celém světě. Jeho použití umožňuje analytikům a vývojářům vytvářet složité dotazy a analyzovat geografická data s vysokou přesností. (OpenStreetMap contributors, 2021)

### Nominatim

Nominatim, jehož název je odvozen z latinského výrazu pro "jmenování", je nástroj a služba pro geokódování, který umožňuje uživatelům vyhledávat OSM data podle názvu nebo adresy a naopak získávat informace o místech z jejich geografických souřadnic. Díky své schopnosti provádět obrácené geokódování a nabízet podrobné výsledky hledání s hierarchií míst je Nominatim cenným nástrojem pro řadu webových služeb a aplikací. Jeho otevřená a přístupná API činí Nominatim atraktivním řešením pro komunitní projekty i komerční využití, kde je potřeba efektivní a přesné geokódování. (OpenStreetMap contributors, 2024)

Všechny tyto nástroje se samozřejmě liší svými schopnostmi, rozhraními a oblastmi použití, ale všechny poskytují silnou podporu pro práci s geoprostorovými daty v rámci ETL procesů.

# 4 Další důležité nástroje a technologie

Následující kapitoly této diplomové práce se budou věnovat klíčovým technologiím, které jsou nezbytné pro úspěšné dokončení praktické části implementace. Podrobně budou rozebrány nejen programovací jazyky potřebné pro vývoj, ale také platforma Grasshopper a možnosti pro efektivní ukládání dat.

## 4.1 Programovací jazyky

Pro manipulaci s geografickými daty se často používají programovací jazyky jako Python, který má knihovny jako GDAL/OGR, Shapely, GeoPandas, a R, který nabízí balíčky jako sp a rgdal. Také Java a Kotlin jsou oba programovací jazyky, které lze použít pro manipulaci s geografickými daty, ale mají různé charakteristiky a využití. Java, dlouholetý jazyk v oblasti vývoje aplikací, je široce používaná pro vývoj serverových aplikací, včetně těch, které pracují s geografickými informačními systémy (GIS). Kotlin, který byl původně navržen jako jazyk pro platformu Android, nabízí moderní syntaxi a je plně kompatibilní s Javou, což umožňuje snadnou integraci do stávajících Java projektů, včetně těch, které se zabývají geografickými daty.

### 4.1.1 Java

Java je silný a flexibilní programovací jazyk, který se široce používá v různých oblastech vývoje softwaru, včetně práce s geografickými daty. Díky své robustní podpoře pro objektově orientované programování a bohatému ekosystému knihoven a nástrojů, Java umožňuje vývojářům efektivně zpracovávat, analyzovat a vizualizovat prostorová data. Geografické informační systémy (GIS) a aplikace založené na poloze využívají Java knihovny, pro práci s mapami, terénními daty a dalšími prostorovými informacemi. Java tak umožňuje vytvářet pokročilé a výkonné aplikace pro prostorovou analýzu, urbanistické plánování, monitorování životního prostředí a další geoprostorové aplikace. Díky své schopnosti integrovat s různými datovými zdroji a podporovat komplexní prostorové analýzy, Java zůstává klíčovou technologií ve světě geoprostorových technologií. Existuje několik Java knihoven, které se dají využít pro práci s GIS a prostorovými daty. Mezi oblíbené knihovny patří:

GeoTools: Open-source knihovna pro Javu nabízející nástroje pro zpracování geoprostorových dat.

JTS Topology Suite (JTS): Knihovna pro Javu určená k tvorbě a manipulaci s vektorovou geometrií.

GeoWave: Java knihovna pro ukládání, indexaci a dotazování geoprostorových dat v distribuovaných systémech.

#### 4.1.1.1 GeoTools

GeoTools je otevřená Java knihovna pro geoprostorové zpracování dat, která podporuje širokou škálu geoprostorových standardů a funkcí. Umožňuje uživatelům efektivně pracovat s různými formáty prostorových dat, provádět analýzy, vizualizace a správu prostorových informací. Knihovna je navržena s důrazem na flexibilitu a rozšiřitelnost, což umožňuje vývojářům snadno integrovat geoprostorové funkce do svých aplikací.

GeoTools je řízeno Výkonným výborem projektu (PMC), který se skládá z dobrovolníků, jež projektu významně přispěli. Projekt využívá otevřený vývojový proces s veřejnou spoluprací na nových nápadech, a nové příspěvky jsou vítány. V roce 2006 se PMC GeoTools rozhodlo připojit k Open Source Geospatial Foundation (OSGeo) a v roce 2007 se stalo jejím oficiálním členem. (Github wiki, 2016)

#### 4.1.1.2 JTS Topology Suite (JTS)

Knihovna JTS Topology Suite (JTS) je open source softwarová knihovna pro Javu, která nabízí objektový model pro planární geometrii společně se základními geometrickými funkcemi. JTS je v souladu s normou Simple Features Specification pro SQL, kterou vydal Open GIS Consortium. JTS je navržena tak, aby sloužila jako klíčová součást vektorově založené geomatické softwaru, jako jsou geografické informační systémy. Lze ji také využít jako univerzální knihovnu poskytující algoritmy v oblasti výpočetní geometrie. (Locationtech Github JTS)

Knihovna JTS Topology Suite (JTS) je robustní nástroj pro práci s geometrií v prostředí Javy, který poskytuje širokou škálu operací a funkcí pro manipulaci s prostorovými daty. Její schopnosti zahrnují podporu různých typů geometrií, jako jsou body, liniové řetězce a polygony, které jsou definovány v rámci specifikace OGC Simple Features pro SQL. (Locationtech Github JTS)

Uživatelé mohou s JTS provádět topologické kontroly platnosti geometrií a získávat základní geometrické informace, jako je plocha a obvod. Knihovna také umožňuje měření vzdálenosti mezi geometriemi a zahrnuje prostorové predikáty založené na modelu Egenhofer DE-9IM, což umožňuje analyzovat vzájemné vztahy mezi geometriemi, jako je překryv, průnik nebo vyloučení. (Locationtech Github JTS)

V oblasti výstupu JTS podporuje formáty jako WKT a GML, což umožňuje snadnou integraci s dalšími geoprostorovými nástroji a systémy. A nakonec, knihovna nabízí také vysokopřesnou aritmetiku pro zajištění správnosti výpočtů, což je nezbytné pro náročné geoprostorové aplikace. (Locationtech Github JTS)

#### 4.1.1.3 GeoWave

GeoWave je open-source knihovna pro ukládání, indexaci a vyhledávání vícerozměrných dat seřazených ve strukturovaných klíč/hodnota úložištích. Podporuje OGC prostorové typy dat až do tří dimenzí a to jak omezené, tak neomezené a časové hodnoty. Podpora jak jednotlivých tak i rozsahových hodnot je zajištěna ve všech dimenzích. GeoWave je postaven na modelu rozšiřitelnosti projektu GeoTools, což umožňuje nativní integraci s jakýmkoliv softwarovým vybavením kompatibilním s GeoTools, například s GeoServerem a UDig, a může zpracovávat data kompatibilní s GeoTools. (Locationtech Github GeoWave, 2022)

Mezi schopnosti GeoWave patří přidání schopnosti indexace vícerozměrných dat do klíč/hodnota úložiště, podpora geografických objektů a geoprostorových operací pro tato úložiště, poskytování pluginu pro GeoServer pro sdílení a vizualizaci geoprostorových dat prostřednictvím standardních OGC služeb a nabízení vstupů a výstupů Map-Reduce pro distribuovanou analýzu a zpracování geoprostorových dat. (Locationtech Github GeoWave, 2022)

### 4.1.2 Kotlin

Kotlin je moderní programovací jazyk, který byl veřejnosti poprvé představen v roce 2016. Jako otevřený zdrojový jazyk Kotlin umožňuje kompilaci do Java bajtkódu, což mu dává flexibilitu být použitelným na různých platformách prostřednictvím Java virtuálního stroje (JVM). Tato kompatibilita se Java rozšiřuje i na knihovny a frameworky, což umožňuje vývojářům snadno integrovat existující Java kód do Kotlin projektů. (Berga, a další, 2023)

Kotlin byl navržen tak, aby byl lepší než Java, s cílem poskytnout čistější a efektivnější syntaxi, rychlejší kompilaci a podporu jak objektově orientovaného, tak i funkcionálního programování. Tyto vlastnosti činí Kotlin atraktivním jazykem pro vývojáře, kteří hledají moderní a efektivní alternativu k Javě. (Berga, a další, 2023)

Kotlin lze efektivně využít pro práci s geografickými daty, jak ukazuje ArcGIS Maps SDK pro Kotlin a WorldWind Kotlin SDK. ArcGIS Maps SDK pro Kotlin poskytuje komplexní podporu pro prostorové reference, které jsou klíčové pro přesnou reprezentaci a manipulaci s

geografickými daty. Umožňuje integraci různých vrstev s odlišnými prostorovými referencemi, podporuje reprojeckci dat za běhu a umožňuje vytváření a zobrazování map s více vrstvami včetně základních map, vrstev prvků a rastrů. SDK také usnadňuje úpravu dat a provádění prostorových analýz tím, že zajistí, že všechny použité geometrie mají známou prostorovou referenci. (Arcgis)

Kromě 2D mapování nabízí WorldWind Kotlin SDK platformu pro vývoj multiplatformních 3D aplikací virtuálního globu. Poskytuje geografický kontext s vysokým rozlišením terénu pro vizualizaci geografických nebo geolokalizovaných informací v 3D i 2D. SDK umožňuje vývojářům přizpůsobit terén a obrazové podklady zeměkoule a podporuje různé tvary pro zobrazování a interakci s geografickými daty. To z něj činí univerzální nástroj pro vytváření aplikací, které vyžadují detailní a interaktivní geografické reprezentace. (WorldWind Community Edition, 2022)

### 4.1.3 Python

Python je moderní programovací jazyk navržený pro snadné psaní programů. Na jedné straně je dostatečně jednoduchý pro začátečníky, ale zároveň nabízí pokročilé nástroje pro složitější projekty. Python má bohatou knihovnu a frameworky, které usnadňují zaměření se na konkrétní problémy bez nutnosti řešit zbytečné podrobnosti. V dnešní době je Python zásadní pro práci s daty díky své schopnosti zpracovávat obrovské datové sady snadno a efektivně. Kniha, která je pravděpodobně zobrazena, je určena každému, kdo se chce naučit, jak využít Python pro analýzu a vizualizaci dat, s důrazem na matematické a statistické operace potřebné pro tuto práci. (PECINOVSKÝ, 2022)

#### 4.1.3.1 Základní práce s Daty

**Čtení a zápis dat**

Python se odlišuje od ostatních jazyků především integrací robustních tříd proudů, které jsou odvozeny z abstraktních základních tříd, a poskytují sadu metod pro efektivní manipulaci s daty. Tyto třídy nabízejí jak univerzální metody pro vstup, tak i pro výstup, což usnadňuje zpracování datových sad různých velikostí a formátů. Zvláštní pozornost je věnována metodám pro správné uzavírání proudů, aby se předešlo potenciálním datovým ztrátám nebo paměťovým únikům. (PECINOVSKÝ, 2022)

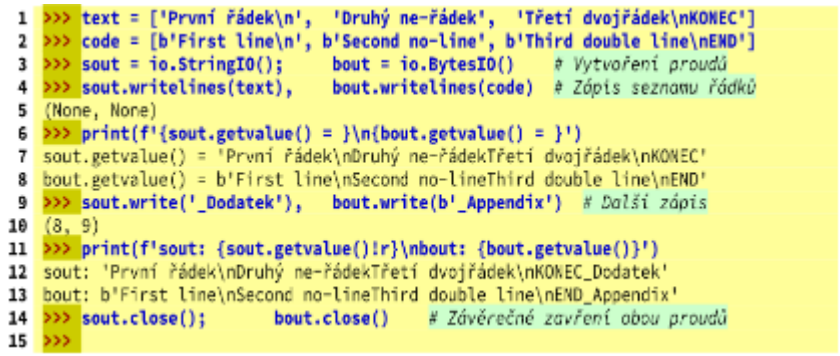
Zásadní částí práce s datovými proudy je čtení dat. V této sekci je potřeba se zaměřit na různé přístupy k čtení dat z proudů, včetně metod `read()`, `readline()`, a `readlines()`. Každá z těchto metod je analyzována s ohledem na jejich specifické použití v kontextu textových a

binárních dat. V ukázce jsou představeny praktické příklady, které ilustrují optimální využití jednotlivých čtecích metod a diskutujeme potenciální výzvy, jako je správné zpracování nových řádků a konce souborů. (PECINOVSKÝ, 2022)

```
>>> # Příprava pomocných proměnných
>>> import io
>>> text = 'První řádek\nDruhý řádek\nTřetí řádek'
>>> code = b'First line\nSecond line\nThird line'   # Pouze ASCII znaky
>>> sio = io.StringIO(text)      # Znakový vstupně-výstupní proud
>>> bio = io.BytesIO (code)      # Binární vstupně-výstupní proud
>>> # Zadání počtu požadovaných řádků nezaručuje, že se opravdu přečtou
>>> sio.readlines(3),   bio.readlines(3)    # Přečte nejvýše tři řádky
(['První řádek\n'], [b'First line\n'])
>>> sio.read(6),        bio.read(6)         # Přečte dalších 6 znaků/bajtů
('Druhý ', b'Second')
>>> # Ukázka načtení binárních dat přímo do vyhrazené paměti
>>> ba = bytearray(b'Start - End')          # Připraví proměnný bajtový objekt
>>> bio.readinto(ba),   ba                  # Načte bajty do vyhrazené paměti
(11, bytearray(b' line\nThird'))
>>> sio.read(),         bio.read()          # Přečte zbytek
('řádek\nTřetí řádek', b' line')
>>> sio.close();        bio.close()         # Na závěr proudy zavře
>>> # Demonstrace proudu jako iterovatelného objektu
>>> def print_lines(stream:'IOBase') -> None:
...     """Vytiskne očíslované řádky daného proudu."""
...     for index, line in enumerate(stream): print(f'{index}. {line!r}')
...
>>> print_lines(io.StringIO(text))  # Argumentem je nově vytvořený proud
0. 'První řádek\n'
1. 'Druhý řádek\n'
2. 'Třetí řádek'
>>> print_lines(io.BytesIO(code))
0. b'First line\n'
1. b'Second line\n'
2. b'Third line'
>>> # Pro demonstraci funkce peek() připravíme obyčejný soubor na disku
>>> from pathlib import Path    # Příští příkaz naplní soubor obsahem
>>> (p := Path('xb.txt')).write_bytes(b'Start - Middle - End'), p.read_bytes()
(20, b'Start - Middle - End')
>>> bx = p.open('br')   # Metoda peek() nezaručuje, kolik toho z proudu načte
>>> print(f'{bx.peek(5)=},    {bx.read()=}')
bx.peek(5)=b'Start - Middle - End',    bx.read()=b'Start - Middle - End'
>>> bx.close()  # Nesmíme zapomenout proud zavřít, aby neblokoval prostředky
>>>
```

*Obrázek 5:Demonstrace použití metod pro čtení. Zdroj: (PECINOVSKÝ, 2022)*

Zápis dat je dalším kritickým aspektem práce s datovými proudy. Sekce podrobně popisuje proces zápisu dat do proudů a zdůrazňuje význam správného uzavírání proudů pro zajištění integrity dat. Jsou prezentovány metody `write()` a `writelines()`, které umožňují zápis jednotlivých řetězců nebo sekvencí řetězců. Příklady zápisu demonstrují jak efektivně pracovat s textovými a binárními daty a jsou diskutovány nejlepší postupy pro zajištění, že zapsaná data jsou odrážkou přesného a zamýšleného výstupu výzkumného pracovníka. (PECINOVSKÝ, 2022)



```
>>> text = ['První řádek\n',  'Druhý ne-řádek',  'Třetí dvojřádek\nKONEC']
>>> code = [b'First line\n', b'Second no-line', b'Third double line\nEND']
>>> sout = io.StringIO();     bout = io.BytesIO()    # Vytvoření proudů
>>> sout.writelines(text),    bout.writelines(code)  # Zápis seznamu řádků
(None, None)
>>> print(f'{sout.getvalue() = }\n{bout.getvalue() = }')
sout.getvalue() = 'První řádek\nDruhý ne-řádekTřetí dvojřádek\nKONEC'
bout.getvalue() = b'First line\nSecond no-lineThird double line\nEND'
>>> sout.write('_Dodatek'),   bout.write(b'_Appendix')  # Další zápis
(8, 9)
>>> print(f'sout: {sout.getvalue()!r}\nbout: {bout.getvalue()}')
sout: 'První řádek\nDruhý ne-řádekTřetí dvojřádek\nKONEC_Dodatek'
bout: b'First line\nSecond no-lineThird double line\nEND_Appendix'
>>> sout.close();       bout.close()    # Závěrečné zavření obou proudů
>>>
```

*Obrázek 6: Demonstrace použití metod pro zápis. Zdroj: (PECINOVSKÝ, 2022)*

Důležité je také uzavírání daných datových proudů. Uzavírání proudů je nezbytné nejen pro prevenci datových ztrát, ale také pro uvolnění systémových zdrojů. Diskutujeme o metodách `close()` a kontextových manažerech s klíčovým slovem `with`, které zajišťují, že proudy jsou správně a bezpečně uzavřeny. Poskytujeme doporučení pro správné uzavírání proudů, aby se minimalizovalo riziko datových nesrovnalostí a zvýšila efektivita využívání zdrojů. (PECINOVSKÝ, 2022)

**Datové třídy**

„Datovou třídu definujeme jako normální třídu, před kterou vložíme dekorátor @daraclass. Tento dekorátor má řadu parametrů, které mají všechny nějaké implicitní hodnoty, takže nemusíte zadávat žádný argument.“ Zmiňuje Pecinovský ve své knize. (PECINOVSKÝ, 2022)

Dekorátor `@dataclass` lze parametrizovat různými argumenty, které řídí chování generovaných metod:

`init`: Řídí vytváření výchozí metody `__init__()`.

`repr`: Určuje, zda je pro třídu generována metoda `__repr__()`.

`eq`: Poskytuje metodu `__eq__()`, která umožňuje porovnávat objekty pro rovnost.

`order`: Pokud je nastaveno na True, jsou poskytnuty další metody pro řazení (`__lt__()`, `__le__()` atd.).

`unsafe_hash`: Pokud je True, třída bude hashovatelná, i když je `eq` True a `frozen` False.

`frozen`: Když je True, objekty jsou po inicializaci neměnné. (PECINOVSKÝ, 2022)

Použití datových tříd může vést k udržitelnějšímu a efektivnějšímu kódu tím, že automatizuje rutinní implementace. To se ukazuje jako zvláště výhodné v datově náročných aplikacích, kde jsou třídy primárně využívány jako kontejnery dat. (PECINOVSKÝ, 2022)

Zavedení datových tříd v Pythonu 3.7 představuje významný krok směrem k efektivnějším programovacím praktikám, zejména pro aplikace zpracovávající velké objemy dat. Modul abstrahuje běžné vzorce v návrhu tříd, což vývojářům umožňuje soustředit se na unikátní aspekty logiky jejich aplikace. (PECINOVSKÝ, 2022)

#### 4.1.3.2 Knihovy v Pythonu

Pro geografické zpracování dat v Pythonu existuje několik klíčových knihoven, které rozšiřují jeho možnosti a umožňují efektivní práci s různými typy geodat. Zde je shrnutí některých z těchto knihoven:

**Arcpy:** Knihovna Arcpy je určena pro geoprocessing operace v prostředí Esri ArcGIS. Umožňuje nejen prostorovou analýzu, ale také konverzi dat, správu a tvorbu map.

**Geopandas:** Geopandas rozšiřuje možnosti knihovny pandas o geografickou složku. Umožňuje pracovat s geografickými daty podobně jako s tabulkovými daty a pro operace jako overlay využívá knihovny Fiona a Shapely.

**GDAL/OGR:** Knihovna GDAL/OGR slouží pro převod mezi různými formáty GIS dat. Je široce využívána v mnoha GIS softvarech pro import a export dat.

**RSGISLib:** RSGISLib je sada nástrojů pro dálkový průzkum a analýzu rastrů. Umožňuje například klasifikaci, filtrování a statistické zpracování obrazových dat.

**PyProj:** Hlavním účelem knihovny PyProj je práce se systémy prostorových referencí. Umožňuje transformaci a projekci souřadnic v různých geografických referenčních systémech.

Pro datovou analýzu a vědecké výpočty Python nabízí také knihovny, jako jsou:

**NumPy:** Knihovna NumPy je základem pro vědecké výpočty v Pythonu. Umožňuje efektivní práci s velkými datovými množinami.

**Matplotlib:** Pro vizualizaci dat, včetně geografických, lze použít knihovnu Matplotlib. Umožňuje kreslení grafů, map a dalších typů vizualizací.

**Pandas:** Pandas je velmi oblíbená knihovna pro manipulaci s daty a jejich analýzu. Je zvláště užitečná pro práci s tabulkovými daty.

**Re (regular expressions):** Pro práci s textovými daty a jejich filtraci lze využít knihovnu pro regulární výrazy. (GISGeography, 2023)

## 4.2 Grasshopper

Grasshopper je vizuální programovací jazyk a prostředí, které funguje uvnitř aplikace pro počítačově podporované navrhování (CAD) Rhinoceros 3D, známé také jako Rhino. Byl vytvořen Davidem Ruttenem ve společnosti Robert McNeel & Associates. Grasshopper umožňuje uživatelům vytvářet programy táhnutím komponent na plátno, což eliminuje potřebu znalosti programování nebo skriptování pro tvorbu generátorů tvarů, což je zvláště užitečné pro designéry pracující s komplexními tvary a povrchy.

Grasshopper 3D je známý tím, že architektům a návrhářům poskytuje výkonnou platformu pro generativní algoritmické modelování, která je bezproblémově integrována s nástroji 3D modelování Rhino. Jeho přitažlivost spočívá v grafickém editoru algoritmů, který uživatelům umožňuje zkoumat nové tvary a návrhy, aniž by potřebovali předchozí znalosti programování nebo skriptování. To umožňuje vytvářet komplexní generátory tvarů, od jednoduchých až po velmi složité návrhy, což z něj činí důležitý nástroj v moderní architektonické a návrhářské praxi. (Davidson, 2024)

Grasshopper je populární mezi studenty i profesionály a je široce používán v architektonickém designu pro jeho intuitivní způsob prozkoumávání návrhů bez nutnosti učit se skriptování. Nabízí mnoho doplňků a je součástí Rhino v8. (Simply Rhino, 2024)

Význam a využití v kontextu této diplomové práce bude dále znázorněn v praktické části této práce.

## 4.3 Databáze - PostgreSQL

Databázové systémy představují klíčovou infrastrukturu pro efektivní správu a analýzu velkých objemů dat, což je nezbytné pro podporu rozhodování a strategického plánování v rámci moderních organizací. Tyto systémy jsou nezbytné v různorodých sektorech díky své schopnosti poskytovat strukturované ukládání a komplexní analýzu dat. S nástupem pokročilých technologií, jako jsou in-memory databáze a cloudová řešení, došlo k významnému posunu v rychlosti a škálovatelnosti databázových operací.

**PostgreSQL v praxi**

Ve světě PostgreSQL se termín "klastr" vztahuje na instanci databáze schopnou spravovat více databází najednou, což zaručuje účinnou správu prostředků a izolaci datových prostorů. Každá databáze v rámci klastru funguje jako samostatná jednotka s vlastními uživateli a přístupovými právy. Uživatelé spojení s jednou databází nemají přístup k datům v ostatních databázích, pokud nejsou explicitně připojeni.

**Organizace databázových objektů**

Organizace objektů v PostgreSQL se provádí skrze schémata, což jsou jmenné prostory umožňující organizaci databázových objektů, jako jsou tabulky a funkce, do logicky strukturovaných skupin. Schémata poskytují užitečný mechanismus pro udržení pořádku a přehlednosti, aniž by byla povolena jejich vnoření, což udržuje organizaci na jednoduché a přehledné úrovni. Každý objekt v databázi je přiřazen k přesně jednomu schématu, a pokud není schéma explicitně specifikováno, předpokládá se výchozí veřejné schéma. Uživatelé jsou definováni na úrovni klastru, což umožňuje správu libovolné databáze, k níž mají přístupová práva. (Ferrari, a další, 2020)

# 5 Praktická část – datový model

Nejdříve se v praktické části diplomové práce zaměříme na jakousi rekapitulaci toho, čeho je potřeba dosáhnout ke splnění cíle diplomové práce. V rámci této praktické části diplomové práce se zaměříme na realizaci specifických úkolů, které vyplývají z teoretických zjištění a analýz provedených v předchozích kapitolách. Hlavním cílem je vyvinout a implementovat efektivní metodiku pro transformaci geografických dat z modelu OpenStreetMap (OSM) do strukturovaného a pro aplikace společnosti snadno využitelného formátu. Tento proces je klíčový pro zlepšení interních systémů společnosti, které závisí na přesnosti a aktuálnosti geolokačních dat.

**Cíle praktické části:**

Praktická část se bude skládat z několika klíčových kroků, které zajistí splnění hlavního cíle diplomové práce:

**Implementace Transformačního Mechanismu**: Vyvinout software, který bude schopen zpracovávat a transformovat data z OSM do interního datového modelu společnosti. Tento mechanismus bude muset řešit různé výzvy, jako jsou rozdíly v datových modelech, nekonzistence ve zdrojových datech a potřeba udržování vysoké úrovně přesnosti dat.

**Optimalizace Procesů**: Návrh a implementace procesů pro pravidelnou aktualizaci dat, což zahrnuje automatizaci extrakčních, transformačních a načítacích procedur, aby se data udržovala aktuální s minimálním lidským zásahem.

**Validace a Verifikace**: Ověření správnosti transformovaných dat porovnáním s očekávanými výstupy a zajištěním, že data splňují požadavky interních stakeholderů a aplikací společnosti.

**Dokumentace a Předání**: Vytvoření ucelené dokumentace k vývojovým a operačním procesům a zajištění, že výsledné řešení je plně předatelné a udržitelné.

**Přínosy implementace projektu**

Úspěšná implementace navrhovaného řešení bude mít značný dopad na efektivitu a rozšíření řešení daného projektu, což otevírá možnosti dalšího využití pro firmu. Rozšířením a zlepšením kvality adresních dat dojde ke zvýšení atraktivnosti nabídky jak pro B2B, tak B2C zákazníky. Vylepšení adresních informací nejenže přispěje k přesnějšímu cílení a personalizaci

služeb, ale také umožní firmě lépe reagovat na tržní požadavky a trendy. Tímto způsobem přispěje diplomová práce nejen k akademickému poznání, ale i k praktickému využití v reálném podnikovém prostředí, což demonstruje významný přesah mezi teoretickým výzkumem a jeho aplikací ve světě podnikání.

## 5.1 Definince datového modelu

V této fázi diplomové práce se zaměříme na důkladnou analýzu a specifikaci požadavků potřebných pro transformaci dat z OpenStreetMap (OSM) do specificky strukturovaného adresního modelu, který je používán v aplikacích jako je Grasshopper. Hluboké pochopení jak struktury zdrojových dat z OSM, tak i detailů cílového datového modelu je nezbytné pro navržení efektivního a přesného procesu mapování dat, který bude schopen adresovat specifické potřeby uživatelů a aplikací společnosti.

OpenStreetMap nabízí jedinečný a rozsáhlý zdroj geografických dat, která jsou výsledkem spolupráce široké komunity. Tato data jsou strukturovaná v komplexním modelu, který zahrnuje uzly (nodes), cesty (ways) a relace (relations). Každý z těchto elementů nese různé atributy a tagy, které popisují geografické a fyzikální charakteristiky objektů na zeměkouli. Pro účely této práce je nezbytné porozumět, jak jsou data organizována a jak mohou být interpretována a transformována do strukturovanějšího a intuitivnějšího formátu.

Na druhé straně, cílový datový model Grasshopper vyžaduje strukturu, která je optimalizovaná pro konkrétní podnikové aplikace, jako jsou logistika, plánování městského rozvoje nebo marketing. Tento model se soustředí na hierarchickou organizaci adresních dat, od jednotlivých adresních bodů až po ulice, části obcí, obce, regiony a státy. Každá úroveň této hierarchie musí být jasně definována a musí obsahovat relevantní atributy potřebné pro specifické aplikace.

Zásadním výstupem analýzy těchto dvou datových modelů bude návrh transformačního procesu, který bude schopen překlenout rozdíly mezi oběma strukturami. Tento proces bude zahrnovat vývoj algoritmů pro extrakci, čištění, transformaci a načítání dat, aby bylo zajištěno, že transformovaná data jsou přesná, konzistentní a aktuální. Tento proces bude rovněž zahrnovat testování transformovaných dat za účelem validace jejich přesnosti a relevanci pro koncové uživatele.

### 5.1.1 Analýza datových struktur zdrojových dat z OpenStreetMap

**OpenStreetMap (OSM) Data Model**:

**Uzly (Nodes)**:

Uzly představují základní geografické body, které mají své specifické zeměpisné souřadnice. Každý uzel může obsahovat doplňkové informace (tagy), které určují jeho vlastnosti nebo charakteristiky. Uzly jsou používány pro reprezentaci konkrétních objektů, jako jsou:

Bodové značky: Ty mohou zahrnovat různé typy infrastruktury a zařízení jako semafory, poštovní schránky, telefonní budky, zastávky veřejné dopravy a další.

Zeměpisné body: Jako jsou vrcholy hor, konkrétní bodové památky, nebo jiné významné geografické lokality.

**Cesty (Ways)**:

Cesty jsou definovány jako uspořádané sekvence uzlů, které tvoří liniové struktury nebo uzavřené polygonální oblasti. Cesty mají mnoho využití, včetně:

Liniové objekty: Jakými jsou ulice, železniční tratě, řeky a další dopravní a přírodní prvky.

Polygonální objekty: Které mohou být budovy, parky, jezera, a další plošné prvky na mapě.

**Relace (Relations)**: Relace jsou pokročilé struktury, které mohou definovat vztahy mezi různými uzly a cestami, a dokonce i mezi jinými relacemi. Relace jsou nezbytné pro popis složitějších vztahů a struktur, jako jsou:

Multipolygonální struktury: Například pro reprezentaci oblastí s dílčími plochami, jako jsou lesy s průsekami nebo vodní plochy s ostrovy.

Administrativní hranice: Pro definici hranic měst, obcí, regionů a států, což umožňuje detailní správu teritoriálních dat.

### 5.1.2 Specifikace požadavků na formát a strukturu dat v cílovém modelu

**Adresní Bod**: Nejmenší jednotka adresy, typicky reprezentuje konkrétní vchod, budovu nebo místo na adrese.

**Ulice**: Shromáždění adresních bodů podél komunikace; každý bod na ulici sdílí název ulice.

**Část Obce**: Specifická oblast města nebo obce, může být použita k dělení velkých městských oblastí.

**Obec**: Město nebo vesnice, základní administrativní jednotka, která shromažďuje části obce a ulice.

**Region (Provincie)**: Vyšší administrativní jednotka, která zahrnuje několik obcí a částí obce.

**Stát**: Nejvyšší úroveň adresní struktury, reprezentuje zemi v rámci které se všechny ostatní jednotky nacházejí.

### 5.1.3 Definice klíčových transformačních pravidel pro mapování dat mezi systémy

**Z Node do Adresního Bodu**: Přiřazení geolokačních dat z uzlů do adresních bodů, včetně přidělení příslušného názvu ulice, čísla domu a dalších atributů relevantních pro adresní bod.

**Z Way do Ulice**: Transformace cest, které představují ulice, do modelu ulice, včetně agregace adresních bodů a definování hranic ulice.

**Z Relations do Částí Obce, Obcí, a Regionů**: Využití relací pro definování a mapování vyšších administrativních a geografických jednotek.

**Agregace Data**: Konsolidace různých zdrojových dat do koherentních jednotek v cílovém modelu, zajistění konzistence a eliminace redundancí.

## 5.2 Výběr a stažení geografických dat

Pro účely této diplomové práce bylo nezbytné zvolit optimální zdroj geografických dat, který by splňoval specifické požadavky projektu a zároveň byl prakticky zpracovatelný na dostupném hardwarovém vybavení. Po důkladném průzkumu dostupných zdrojů bylo rozhodnuto využít data poskytovaná portálem (https://download.geofabrik.de/europe.html), který nabízí rozsáhlé geografické informace rozdělené podle jednotlivých regionů a zemí.

**Důvody výběru regionálních dat**

Vzhledem k následujícím důvodům byl výběr omezen na regionální data, konkrétně na oblasti v Evropě:

1. Praktičnost: Celosvětová data OpenStreetMap jsou obrovská a jejich zpracování by na běžně dostupných počítačích trvalo nepřiměřeně dlouho a vyžadovalo by významné hardwarové zdroje, které nebyly k dispozici.
2. Relevance: Požadavky projektu specifikovaly zaměření na analýzu a testování dat z geografických oblastí blízkých České republice. Proto bylo logické vybrat pouze data těch zemí, které jsou pro projekt nejrelevantnější.
3. Efektivita testování: Stahování a práce s menšími, regionálně ohraničenými datovými sadami umožňuje efektivnější testování a iteraci, což je klíčové pro rychlou adaptaci a zlepšování transformačních procesů.

Zvolené geografické soubory pro regiony jako jsou Česká republika a sousední státy byly staženy ve formátu `.osm.pbf`.

```
<node id="26862607" version="6" timestamp="2021-12-10T21:29:32Z" lat="43.3272887" lon="18.0167964">
  <tag k="ele" v="1969"/>
  <tag k="name" v="Botin"/>
  <tag k="name:sr" v="Ботин"/>
  <tag k="natural" v="peak"/>
</node>
<node id="26862647" version="3" timestamp="2021-12-09T00:50:45Z" lat="44.2863889" lon="16.2263889">
  <tag k="name" v="Bursaci"/>
  <tag k="name:sr" v="Бурсаци"/>
  <tag k="name:sr-Latn" v="Bursaci"/>
  <tag k="natural" v="peak"/>
</node>
<node id="26862745" version="6" timestamp="2023-09-19T19:06:39Z" lat="42.9704299" lon="17.7527543">
  <tag k="ele" v="955"/>
  <tag k="name" v="Crkvina"/>
  <tag k="name:hr" v="Crkvina"/>
  <tag k="natural" v="peak"/>
</node>
<node id="26862748" version="7" timestamp="2023-08-26T11:17:43Z" lat="44.6006688" lon="16.5267632">
  <tag k="ele" v="1605"/>
  <tag k="name" v="Crni vrh"/>
  <tag k="name:sr" v="Црни врх"/>
  <tag k="natural" v="peak"/>
</node>
<node id="26862781" version="8" timestamp="2019-10-03T07:56:44Z" lat="43.3205156" lon="19.0858569">
  <tag k="alt_name" v="Dernjačista"/>
  <tag k="ele" v="2238"/>
  <tag k="name" v="Dernečište"/>
  <tag k="name:sr" v="Дернечиште"/>
  <tag k="natural" v="peak"/>
  <tag k="prominence" v="1247"/>
  <tag k="wikidata" v="Q11693963"/>
</node>
<node id="26862786" version="7" timestamp="2022-08-13T21:54:47Z" lat="44.4080748" lon="16.9207416">
  <tag k="ele" v="1483"/>
  <tag k="name" v="Mali Dimitor"/>
  <tag k="name:bs" v="Mali Dimitor"/>
  <tag k="name:hr" v="Mali Dimitor"/>
  <tag k="name:sr" v="Мали Димитор"/>
  <tag k="natural" v="peak"/>
</node>
```

*Obrázek 7: ilustrační data Bosny a Hercegoviny ve formáte .osm. Zdroj: (Geofabrik GmbH, a další, 2018)*

## 5.3 Příprava testovacího prostředí

Pro účely validace a testování transformačních procesů dat bylo klíčové vytvořit stabilní a reprezentativní testovací prostředí, které by co nejvíce odpovídalo produkčnímu prostředí používanému ve firmě. To zahrnovalo instalaci a konfiguraci databázového systému PostgreSQL ve verzi 15, která je aktuálně využívána i v podnikovém prostředí společnosti.

**Instalace PostgreSQL a rozšíření**

Vzhledem k tomu, že adresní data vyžadují specifické zpracování a analýzu prostorových informací, bylo rozhodnuto rozšířit standardní instalaci PostgreSQL o následující komponenty:

PostGIS: Toto rozšíření bylo instalováno za účelem podpory pokročilých prostorových a geografických dotazů. PostGIS je nezbytný pro efektivní práci s geolokačními daty, umožňuje provádění komplexních dotazů, jako jsou prostorové spojení, hledání bodů v polygonu, a další geografické operace.

HStore: Rozšíření HStore bylo implementováno pro flexibilní práci s neuspořádanými datovými sadami, což umožňuje uložení a dotazování párů klíč-hodnota přímo v rámci tabulek PostgreSQL. Toto rozšíření je zvláště užitečné pro manipulaci s dynamickými daty, která jsou typická pro OpenStreetMap.

**Konfigurace databáze**

Databázové prostředí bylo konfigurováno tak, aby optimalizovalo výkon pro operace specifické pro zpracování a analýzu prostorových dat. To zahrnovalo nastavení vhodných indexů, úpravy konfiguračních parametrů databáze pro zvýšení rychlosti dotazů a implementaci procedur pro pravidelnou údržbu databáze, jako je VACUUM a ANALYZE, které pomáhají udržovat databázi v optimálním stavu.

## 5.4 Volba Nástrojů pro prvotní transformaci dat

V průběhu přípravné fáze projektu bylo zjištěno, že data poskytovaná platformou OpenStreetMap jsou ve své původní formě extrémně rozsáhlá a komplexní, což představuje významné výzvy v kontextu jejich transformace a integrace do cílového systému. Kromě toho byla tato data často neprovázaná a neuspořádaná z hlediska potřeb specifického podnikového využití. Tato realita nás vedla k rozhodnutí využít existující nástroje pro prvotní transformaci dat, aby se zefektivnil proces a zredukovaly nároky na vývoj vlastních transformačních algoritmů od základů.

### Použití nástroje OSM2PGSQL

Prvním testovaným řešením byl nástroj OSM2PGSQL, který je široce uznáván pro svou schopnost efektivně převádět data z formátu OSM do PostgreSQL databáze s PostGIS rozšířením. Tento nástroj poskytuje robustní framework pro načítání dat a jejich prostorové indexace, což umožňuje snazší manipulaci a dotazování. Avšak, i přes jeho silné stránky, jsme zjistili, že pro specifické požadavky našeho projektu bylo potřeba dalšího zpracování a úpravy dat.

### Přechod na Nominatim

V další fázi jsme se rozhodli vyzkoušet Nominatim, sofistikovanější nástroj speciálně navržený pro geokódování a reverzní geokódování adres z dat OSM. Nominatim nabízí rozsáhlé možnosti pro práci s adresními informacemi, včetně konverze surových geodat na strukturovanější a lépe organizované formáty vhodné pro naše využití. Tento nástroj nám umožnil lépe adresovat požadavky na přesnost a relevanci adresních dat v rámci systému společnosti. Především dokázal data efektivně provazovat mezi různými entitami.

# 6 Prvotní transformace – využití nástroje Nominatim

V rámci diplomové práce byl pro prvotní transformaci geografických dat zvolen nástroj Nominatim, který je přístupný jako open-source projekt na GitHubu a v našem případě je provozován prostřednictvím Docker kontejnerů. Tento přístup umožňuje efektivní a flexibilní nasazení geokódovacího nástroje s minimálními nároky na konfiguraci a údržbu systému.

## 6.1 Volba nástoje

Volba nástroje Nominatim pro práci s geografickými daty byla motivována několika klíčovými faktory. Jedním z hlavních důvodů byla snadná implementace; díky možnosti spuštění Nominatimu v Docker kontejneru je možné rychle nasadit tento nástroj bez nutnosti složitého nastavování serverů a závislostí. Kromě toho Nominatim disponuje širokou podporou a je dobře dokumentován na GitHubu, což usnadňuje orientaci v jeho funkcionalitách a možnostech. V neposlední řadě je Nominatim kompatibilní s existujícími systémy firmy, zejména s databázovými technologiemi PostgreSQL a PostGIS, což umožňuje jeho efektivní integraci do stávající infrastruktury.

## 6.2 Implementace

Požadavkem firmy bylo, aby Nominatim byl schopen fungovat v multi-regionálním režimu a zároveň pracovat s externí databází, což umožňuje centralizované správání a zpracování dat napříč různými geografickými oblastmi. Tento požadavek byl kritický pro integraci a aktualizaci geografických dat ve větším a komplexnějším prostředí než nabízí standardní lokální databáze používaná v běžných instalacích Dockeru. Cílem bylo zajistit, že data budou importována a spravována v externí PostgreSQL databázi, což přináší výhody v podobě lepšího výkonu, vyšší dostupnosti a lepších možností zálohování a obnovy. Pro dosažení tohoto cíle bylo nezbytné provést úpravy v konfiguraci Docker kontejneru a implementovat řešení, které umožňuje efektivní komunikaci s externí databází.

Bez této konfigurace pracuje totiž Nominatim ve spojení s Dockerem pouze s jedním na počátku definovaným regionem, který již později není možno nějak upravovat. A pracuje s interní databází, která by pro naše použití byla přebytečná, jelikož by takto data duplikovala.

### 6.2.1 Stažení a příprava Docker obrazu(image)

Začali jsme stažením nejnovější verze Nominatimu z GitHub repozitáře, který byl k dispozici jako Docker image. Toto řešení nám umožnilo získat předkonfigurované prostředí s

všemi potřebnými závislostmi a aplikacemi, což výrazně zjednodušilo další kroky instalace a nastavení. Některé přednastavené věci zde ovšem byly zbytečné, takže ty byly potřeba odstranit a nahradit tím, co splňovalo naše požadavky.

### 6.2.2 Fungování Dockerfile a jeho role ve vytváření Docker image

Dockerfile je textový dokument, který obsahuje všechny příkazy a instrukce potřebné k vytvoření Docker obrazu, což je vlastně šablona pro Docker kontejnery.

**Struktura a Funkce Dockerfile**:

Základní obraz: FROM ubuntu:jammy AS build určuje základní obraz, od kterého proces sestavení začíná. Tato direktiva nastavuje základní vrstvu Docker image na specifickou verzi operačního systému Ubuntu.

Environmentální proměnné: ENV nastavuje environmentální proměnné jako DEBIAN_FRONTEND a LANG, které ovlivňují chování softwaru během instalace a běhu.

Pracovní adresář: WORKDIR definuje pracovní adresář kontejneru, kde se budou vykonávat všechny následující instrukce.

Instalace balíčků: RUN spouští příkazy v shellu kontejneru. V tomto případě instaluje potřebné závislosti a nástroje potřebné pro sestavení a běh Nominatim.

Konfigurace služeb: Další RUN příkazy konfigurují PostgreSQL a další součásti systému, jako jsou listen adresy a přístupové pravidla.

Instalace aplikace: Poslední RUN příkaz zajišťuje stažení, kompilaci a instalaci Nominatim aplikace.

Kopírování konfiguračních souborů: COPY příkazy kopírují konfigurační soubory a skripty do obrazu, což umožňuje konfiguraci služeb a jejich automatické spuštění.

Zveřejnění portů: EXPOSE informuje Docker o tom, že kontejner bude naslouchat na specifikovaných síťových portech.

Startovací příkaz: CMD specifikuje výchozí příkaz, který se spustí při startu kontejneru.

**Význam Dockerfile a Docker image**:

Dockerfile představuje soubor "receptů" pro vytvoření izolovaného prostředí aplikace, které je nezávislé na hostitelském operačním systému. Když je Dockerfile dokončen a obraz

sestaven, může být Docker image nahrán na registru jako Docker Hub, odkud si jej mohou stáhnout uživatelé a spustit na svém vlastním hardware bez nutnosti dalšího nastavování.

### 6.2.3 Konfigurace Docker kontejneru

**Regionální data a aktualizace:**

Původní skripty a Dockerfile byly upraveny tak, aby podporovaly stahování a aktualizace dat pro více regionů najednou. Například do Dockerfilu byla potřeba doplnit utilita Wget, která tato stahování podporuje. Dále toto bylo realizováno přidáním podpory pro dynamické určení zemí a regionů v rámci skriptů, jako je import_multiple_regions.sh a update_database.sh. Tyto skripty nyní umožňují specifikovat seznam zemí a stahovat jejich odpovídající OSM data pro import nebo aktualizace.

#### 6.2.3.1 Import

import_multiple_regions.sh – Poznámkový blok

Soubor Úpravy Formát Zobrazení Nápověda

```
BASEURL="https://download.geofabrik.de"
DOWNCOUNTRYPOSTFIX="-latest.osm.pbf"

# End of configuration section
# ********************************************************************************


UPDATEDIR=update

IMPORT_CMD="nominatim import"

mkdir -p ${UPDATEDIR}
pushd ${UPDATEDIR}
rm -rf tmp
mkdir -p tmp
popd

for COUNTRY in $COUNTRIES;
do
    echo "======================================================================"
    echo "$COUNTRY"
    echo "======================================================================"
    DIR="$UPDATEDIR/$COUNTRY"
    DOWNURL="$BASEURL/$COUNTRY$DOWNCOUNTRYPOSTFIX"
    IMPORTFILE=$COUNTRY$DOWNCOUNTRYPOSTFIX
    IMPORTFILEPATH=${UPDATEDIR}/tmp/${IMPORTFILE}

    touch2 $IMPORTFILEPATH
    wget ${DOWNURL} -O $IMPORTFILEPATH

    touch2 ${DIR}/sequence.state
    sudo pyosmium-get-changes -O $IMPORTFILEPATH -f ${DIR}/sequence.state -v

    IMPORT_CMD="${IMPORT_CMD} --osm-file ${IMPORTFILEPATH}"
    echo $IMPORTFILE
    echo "======================================================================"
done
```

Řádek 1, Sloupec 1 | 100 % | Unix (LF) | UTF-8

*Obrázek 8: Script import_multiple_regions.sh. Zdroj: vlastní zpracování*

Skript `import_multiple_regions.sh` je navržen tak, aby automatizoval proces stahování a importu geografických dat pro různé regiony do databáze Nominatim. Tady je popis jeho funkce:

Na začátku skript definuje základní URL (`BASEURL`) ze které budou data stahována, v tomto případě z webu Geofabrik, který poskytuje OSM data rozdělená podle regionů. Každý soubor je označen postfixem `-latest.osm.pbf`, což označuje, že se jedná o nejnovější verzi dat ve formátu PBF.

Skript dále nastavuje proměnnou `UPDATEDIR`, což je pracovní adresář, kam budou dočasná data uložena. Tento adresář je připraven a staré dočasné soubory jsou odstraněny pro čistý start procesu. Po proběhnutí importu jsou tyto soubory opět odstraněny, aby se zabránilo jejiž zbytečnému zabírání paměti uvnitř kontejneru.

Pro každou zemi nebo region specifikovaný v proměnné `COUNTRIES` provádí skript následující kroky:

1. Vytvoření pracovního adresáře pro region: Pro každý region vytvoří příslušný podadresář v `UPDATEDIR`.

2. Stahování OSM dat: Pomocí nástroje `wget` skript stáhne aktuální OSM data pro daný region a uloží je do dočasného pracovního adresáře.

3. Příprava na aktualizaci dat: Skript inicializuje soubor `sequence.state`, který je potřebný pro následné aktualizace a uchovává informace o posledním stavu sekvenčního čísla pro každý region.

4. Aktualizace dat: Pomocí nástroje `pyosmium-get-changes` skript získá poslední změny a aktualizuje soubor OSM dat. Toto se provádí kvůli následnému importu, jelikož je zde potřeba informace o poslední správné verzi pro aktualizační soubor.

5. Import dat do Databáze: Příkazem `nominatim import` se spustí import aktualizovaných OSM dat do Nominatim databáze.

Po zpracování všech regionů skript vypíše zprávu o dokončení importu a uklidí za sebou dočasné soubory, aby uvolnil prostor a připravil prostředí pro další použití.

#### 6.2.3.2 Update

```
UPDATEBASEURL="https://download.geofabrik.de"
UPDATECOUNTRYPOSTFIX="-updates/"

UPDATEDIR="update"

for COUNTRY in $COUNTRIES;
do
    echo "======================================================================="
    echo "$COUNTRY"
    echo "======================================================================="
    DIR="$UPDATEDIR/$COUNTRY"
    FILE="$DIR/sequence.state"
    BASEURL="$UPDATEBASEURL/$COUNTRY$UPDATECOUNTRYPOSTFIX"
    FILENAME=${COUNTRY//[\/]/_}

    echo "Attempting to get changes"
    rm -f ${DIR}/${FILENAME}.osc.gz
    pyosmium-get-changes -o ${DIR}/${FILENAME}.osc.gz -f ${FILE} --server $BASEURL -v

    echo "Attempting to import diffs"

    nominatim add-data --diff ${DIR}/${FILENAME}.osc.gz
done
```

*Obrázek 9: Update_database.sh script. Zdroj: Vlastní zpracování*

V předchozí části byl popsán proces importu více regionů pomocí skriptu import_multiple_regions.sh. Nyní se zaměříme na skript update_database.sh, jehož cílem je průběžně aktualizovat data již naimportovaných regionů v Nominatim databázi. Klíčovým rozdílem oproti skriptu pro import je způsob, jakým update_database.sh zachází s daty -

namísto kompletního stahování a importu datových souborů se soustředí na inkrementální aktualizace.

Skript update_database.sh využívá soubor sequence.state pro každý region, což je mechanismus pro sledování poslední úspěšně stažené aktualizace. Tento soubor uchovává sekvenční číslo, které označuje verzi dat, od které se má stáhnout následující sada změn. Umožňuje tím zajistit, že aktualizace jsou stahovány a aplikovány sekvenčně a v pořádku, čímž se předejde možnému poškození dat způsobeného skoky nebo duplikací aktualizací.

Postup skriptu je následující:

Stahování změn: Skript prochází seznamem konfigurovaných regionů a pro každý region se pokusí stáhnout nejnovější změny dat. Tyto změny jsou staženy ve formě souborů .osc.gz a jsou získány pomocí nástroje pyosmium-get-changes, který využívá informace ze souboru sequence.state.

Import rozdílových dat: Po úspěšném stažení souborů se změnami skript spustí nominatim add-data k naimportování těchto dat do databáze. Tento přístup je efektivnější, protože nevyžaduje import celého datasetu, ale pouze nově příchozí změny.

Aktualizace poštovních kódů a indexace: Po naimportování změn skript spustí příkaz nominatim refresh --postcodes k aktualizaci poštovních kódů v databázi. Následně se spustí indexace pomocí nominatim index --threads $THREADS, což zajistí, že nově přidaná data jsou správně indexována a vyhledatelná.

Na konci skriptu je uživateli oznámeno dokončení procesu a databáze je připravena na další použití s nejnovějšími aktualizovanými daty.

#### 6.2.3.3 Přizpůsobení konfigurace:

Konfigurační soubory a skripty, jako config.sh, byly upraveny, aby reflektovaly potřeby externí databáze a zvláštní požadavky na výkon a bezpečnost. Změny zahrnovaly přizpůsobení nastavení replikace, úpravy PostgreSQL konfigurace pro lepší výkon (např. shared_buffers, maintenance_work_mem,…), a podmínky pro import stylů a doplňkových datových souborů.

#### 6.2.3.4 Integrace s externí databází:

Byla přidána podpora pro práci s externí PostgreSQL databází, což zahrnuje konfigurační úpravy v Docker kontejneru pro správné připojení a autentizaci. Tato změna umožňuje Nominatimu pracovat s daty uloženými mimo lokální prostředí Dockeru, což je klíčové pro podporu podnikových IT politik.

#### 6.2.3.5 Automatizace a skriptování:

Pro zjednodušení managementu a automatizaci rutinních úkolů byly vytvořeny nebo upraveny skripty, jako start_update_stop.sh, které umožňují automatizované spouštění aktualizací, zastavování a restartování kontejnerů. Tyto skripty zjednodušují pravidelnou údržbu a aktualizace dat bez manuálního zásahu.

### 6.2.4 Testování běhu Docker kontejneru

**Přizpůsobení Docker image a jeho nastartování:**

Změny v Docker image zahrnovaly úpravy ve skriptech jako import_multiple_regions.sh a update_database.sh, což umožňuje importovat a aktualizovat více regionů najednou. Pro nastavení a spuštění Docker kontejneru s přizpůsobenými proměnnými prostředí bylo použito následující příkazové volání:

```
docker run -d -e NOMINATIM_DATABASE_DSN="pgsql:dbname=nominatim;host=host.docker.internal;user=postgres;password=4848" -e PGHOST=host.docker.internal -e PGDATABASE=nominatim -e PGUSER=postgres -e PGPASSWORD=4848 -e IMPORT_STYLE=address --name nominatim my-nominatim
```

V tomto příkazu -d (detached mode) znamená, že kontejner poběží na pozadí a nezablokuje příkazovou řádku. Proměnné prostředí konfigurují přístup k databázi a upřesňují styl importu pro Nominatim, což zajišťuje, že kontejner bude komunikovat s určenou externí databází. Připojení do databáze je zde v ukázce pro lokální databázi.

**Provedení importu:**

Po úspěšném spuštění kontejneru byl proveden import dat pomocí skriptu, který byl součástí upraveného Docker image. Proces byl iniciován následujícím příkazem:

```
docker exec -it nominatim /app/import_multiple_regions.sh
```

Tento příkaz spouští shell skript import_multiple_regions.sh, který se stará o stahování a nahrávání dat pro určené regiony. Jak bylo popsáno v kapitole 6.2.2 Skript byl pečlivě testován, aby bylo zajištěno, že data jsou správně importována a v souladu s očekáváními uživatelů. Zvláštní pozornost byla věnována ověření, že po dokončení importu lze kontejner bez problémů zastavit, aby uvolnil systémové zdroje.

**Provedení aktualizace:**

Pro testování aktualizací bylo možné využít buď přiloženého dávkového souboru start_update_stop.bat nebo ekvivalentního shell skriptu start_update_stop.sh, v závislosti na použitém operačním systému. Jelikož po importu dat byl kontejner zastaven, je proto v těchto scriptech také jeho opětovné rozběhnutí před importem a zastavení po importu aktualizačních dat. Pro pravidelné spouštění těchto aktualizací je pak dále nutno nastavit běhy těchto skriptů prostřednictvím Task Scheduleru ve windows nebo cron jobu v linuxu.

**Přidání dat do existující databáze – kontejneru**

V rámci testování běhu Docker kontejneru bylo klíčové ověřit, že Nominatim umožňuje přidání dat do této databáze bez nutnosti restartování nebo vytváření nové instance kontejneru. Cílem bylo zjistit, jak efektivně lze rozšířit databázi o další regiony a jak se tyto nové informace projeví v celkové funkčnosti systému. Tato funkčnost byla kritická, jelikož firma potřebovala flexibilitu v importu dat z různých regionů a jejich pravidelné aktualizace.

Pro import nových dat byly použity následující příkazy:

nominatim add-data --file https://download.geofabrik.de/europe/monaco-latest.osm.pbf - Tento příkaz zahájí proces importu nejnovějších dat OSM pro zvolený region, v tomto případě Monako.

nominatim refresh --postcodes - Po úspěšném importu je nutné aktualizovat informace o poštovních směrovacích číslech, aby odrážely nejnovější změny obsažené v nových datech.

nominatim index -j 6 - Indexace je posledním krokem procesu, kde -j 6 určuje, že proces by měl využít 6 vláken pro paralelní zpracování, což zvyšuje rychlost indexace.

Tyto příkazy se dají zapisovat buďto v rámci Docker aplikace nebo pomocí příkazového volání začínajícího docker exec název_kontejneru.

Proces aktualizace dat pro konkrétní region po jejich přidání přes add-data vyžaduje jiný přístup, jelikož skript update_database.sh není navržen pro opakované spouštění aktualizací pro stávající kontejner. Místo toho je třeba nastavit REPLICATION_URL pomocí příkazu:

REPLICATION_URL=https://download.geofabrik.de/europe/monaco-latest.osm.pbf

A následně provést inicializaci a jednorázovou aktualizaci dat pomocí příkazů:

nominatim replication --init pro inicializaci prostředí pro replikaci.

nominatim replication --once pro provedení jednorázové aktualizace.

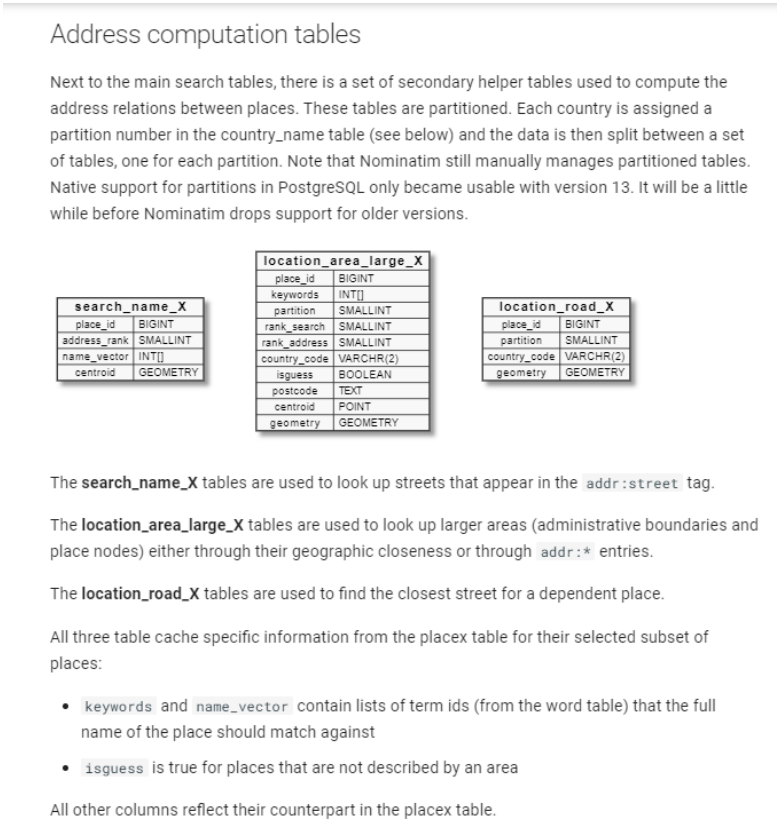
Tento přístup vyžaduje manuální nastavení každého restartu Nominatim, což může být časově náročné a může zvyšovat riziko chyb. Proto se doporučuje, že pro jednoduchost a stabilitu je preferováno využívat pouze původně naimportované země a omezit frekvenci aktualizací, případně tento proces automatizovat pomocí Task Scheduleru nebo cron jobu, pokud je potřeba provádět pravidelné aktualizace. Tedy je pouze taková možnost, která se dá využít v krajních případech dočasného fungování dané databáze.

## 6.3 Výsledky Nominativu – datový model

Jak již bylo zmíněno, klíčovým výstupem procesu importu dat pomocí Nominatimu bylo vytvoření a naplnění externí databáze strukturovanými daty. Po proběhnutí byla pak následně provedena indexace, což je proces, při kterém Nominatim vytváří prostorové a textové indexy, které umožňují rychlé vyhledávání a dotazování. Tato fáze je kritická pro zajištění efektivity následného provazování mezi entitatami a geokódování v databázi.

### 6.3.1 Výpočetní tabulky adres

Nominatim vytváří mnoho tabulek, které jsou klíčové pro úspěšný import a správnou funkčnost celého systému. Tyto tabulky uchovávají data o všem od jednotlivých bodů až po složité relace. Hlavní tabulkou, kde jsou všechny důležité informace soustředěny, je placex. Tato tabulka slouží jako centrální bod pro veškeré dotazování a je základem pro většinu operací vyhledávání a geokódování v Nominatimu. Tato struktura nejen zlepšuje výkon dotazů, ale také zjednodušuje správu databáze tím, že umožňuje operace jako jsou aktualizace, změny a údržba na menších segmentech dat místo celkové datové sady.



## Address computation tables

Next to the main search tables, there is a set of secondary helper tables used to compute the address relations between places. These tables are partitioned. Each country is assigned a partition number in the country_name table (see below) and the data is then split between a set of tables, one for each partition. Note that Nominatim still manually manages partitioned tables. Native support for partitions in PostgreSQL only became usable with version 13. It will be a little while before Nominatim drops support for older versions.

**search_name_X**

| | |
|---|---|
| place_id | BIGINT |
| address_rank | SMALLINT |
| name_vector | INT[] |
| centroid | GEOMETRY |

**location_area_large_X**

| | |
|---|---|
| place_id | BIGINT |
| keywords | INT[] |
| partition | SMALLINT |
| rank_search | SMALLINT |
| rank_address | SMALLINT |
| country_code | VARCHR(2) |
| isguess | BOOLEAN |
| postcode | TEXT |
| centroid | POINT |
| geometry | GEOMETRY |

**location_road_X**

| | |
|---|---|
| place_id | BIGINT |
| partition | SMALLINT |
| country_code | VARCHR(2) |
| geometry | GEOMETRY |

The **search_name_X** tables are used to look up streets that appear in the `addr:street` tag.

The **location_area_large_X** tables are used to look up larger areas (administrative boundaries and place nodes) either through their geographic closeness or through `addr:*` entries.

The **location_road_X** tables are used to find the closest street for a dependent place.

All three table cache specific information from the placex table for their selected subset of places:

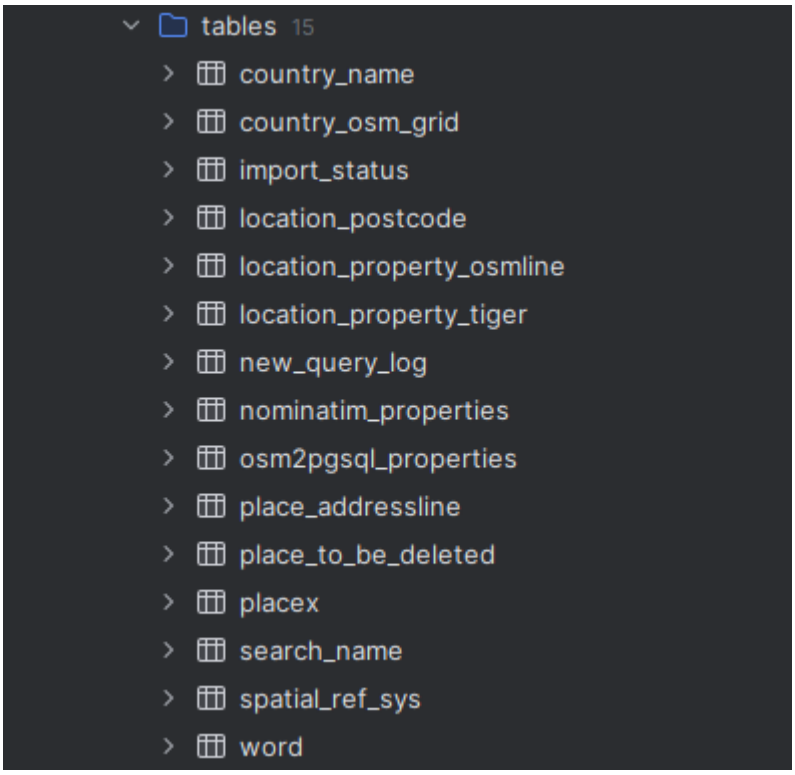
- `keywords` and `name_vector` contain lists of term ids (from the word table) that the full name of the place should match against
- `isguess` is true for places that are not described by an area

All other columns reflect their counterpart in the placex table.

*Obrázek 10: tabulky potřebné pro indexaci. Zdroj: (Nominatim developer community, 2020)*

Ve firmě byla položena otázka, zda jsou tyto pomocné tabulky nezbytně nutné pro běžný provoz databáze. Příkaz nominatim freeze je možné použít pro odstranění těchto tabulek a převod databáze do 'read-only' režimu. Toto má za následek optimalizaci využití diskového prostoru a může zlepšit výkon databáze pro dotazování, ale zároveň to znemožňuje další aktualizace dat. Jakmile je databáze 'zmrazena', nové informace již nemohou být importovány ani existující data nemohou být aktualizovány.

Pro firmy, které potřebují databázi pouze pro čtení a neplánují provádět další aktualizace, může být tato možnost přijatelným kompromisem. V případě, že je však požadována dynamická databáze s průběžnými aktualizacemi, musí se zachovat existující struktura tabulek, aby byly zachovány schopnosti aktualizace a údržby dat




*Obrázek 11: tabulky Nominatimu po "zmrazení". Zdroj: Vlastní zpracování*

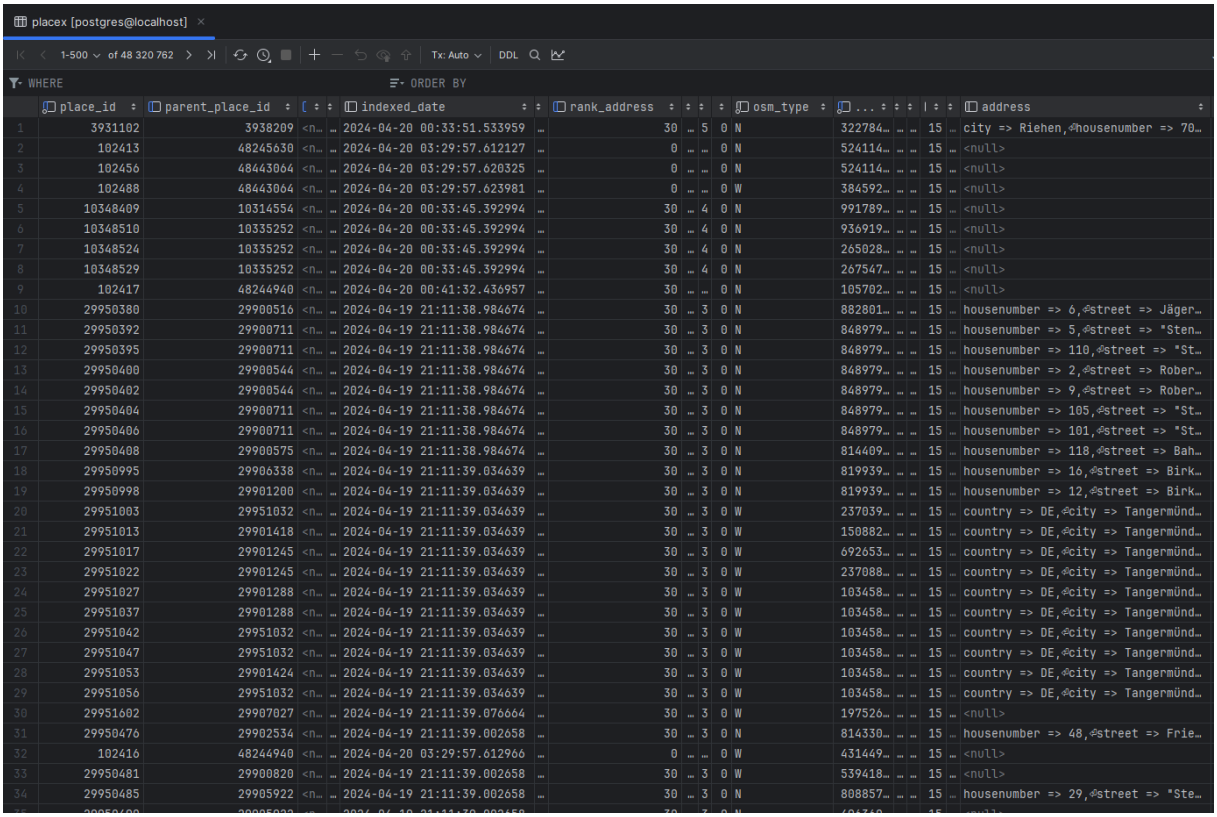
### 6.3.2 Samotná data

Centrálním prvkem datového modelu Nominatimu je tabulka placex, která představuje páteřní strukturu pro všechna geografická data. Tato tabulka obsahuje klíčové sloupce place_id a parent_place_id, které hrají zásadní roli v organizaci a propojení jednotlivých geografických entit.

Sloupec place_id je jedinečný identifikátor přiřazený každé položce v tabulce, což umožňuje jednoznačné odlišení mezi různými objekty, jako jsou adresní místa, ulice či celé městské části. Sloupec parent_place_id pak slouží k definování hierarchických vztahů mezi entitami, kde například adresa může mít jako rodičovské ID identifikátor ulice nebo čtvrti, v níž se nachází. Tímto způsobem je možné vybudovat složité hierarchie a vztahy, jako jsou například adresa patřící do určité ulice, která je součástí širší městské části a nakonec spadá do konkrétního města nebo regionu.

Díky tomuto provázání může Nominatim poskytovat odpovědi na komplexní geografické dotazy. Může například snadno identifikovat všechny adresy v rámci určité ulice nebo vyhledat všechny ulice patřící do dané městské části. Tento přístup nejenže zjednodušuje dotazování a umožňuje efektivní vyhledávání v databázi, ale také umožňuje komplexní analýzy

a geografické dotazy, které jsou klíčové pro široké spektrum aplikací od navigace až po plánování města.



| | place_id | parent_place_id | indexed_date | rank_address | osm_type | address |
|---|---|---|---|---|---|---|
| 1 | 3931102 | 3938209 | 2024-04-20 00:33:51.533959 | 30 | N | city => Riehen,housenumber => 70… |
| 2 | 102413 | 48245630 | 2024-04-20 03:29:57.612127 | 0 | N | <null> |
| 3 | 102456 | 48443064 | 2024-04-20 03:29:57.620325 | 0 | N | <null> |
| 4 | 102488 | 48443064 | 2024-04-20 03:29:57.623981 | 0 | W | <null> |
| 5 | 10348409 | 10314554 | 2024-04-20 00:33:45.392994 | 30 | N | <null> |
| 6 | 10348510 | 10335252 | 2024-04-20 00:33:45.392994 | 30 | N | <null> |
| 7 | 10348524 | 10335252 | 2024-04-20 00:33:45.392994 | 30 | N | <null> |
| 8 | 10348529 | 10335252 | 2024-04-20 00:33:45.392994 | 30 | N | <null> |
| 9 | 102417 | 48244940 | 2024-04-20 00:41:32.436957 | 30 | N | <null> |
| 10 | 29950380 | 29900516 | 2024-04-19 21:11:38.984674 | 30 | N | housenumber => 6,street => Jäger… |
| 11 | 29950392 | 29900711 | 2024-04-19 21:11:38.984674 | 30 | N | housenumber => 5,street => "Sten… |
| 12 | 29950395 | 29900711 | 2024-04-19 21:11:38.984674 | 30 | N | housenumber => 110,street => "St… |
| 13 | 29950400 | 29900544 | 2024-04-19 21:11:38.984674 | 30 | N | housenumber => 2,street => Rober… |
| 14 | 29950402 | 29900544 | 2024-04-19 21:11:38.984674 | 30 | N | housenumber => 9,street => Rober… |
| 15 | 29950404 | 29900711 | 2024-04-19 21:11:38.984674 | 30 | N | housenumber => 105,street => "St… |
| 16 | 29950406 | 29900711 | 2024-04-19 21:11:38.984674 | 30 | N | housenumber => 101,street => "St… |
| 17 | 29950408 | 29900575 | 2024-04-19 21:11:38.984674 | 30 | N | housenumber => 118,street => Bah… |
| 18 | 29950995 | 29906338 | 2024-04-19 21:11:39.034639 | 30 | N | housenumber => 16,street => Birk… |
| 19 | 29950998 | 29901200 | 2024-04-19 21:11:39.034639 | 30 | N | housenumber => 12,street => Birk… |
| 20 | 29951003 | 29951032 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 21 | 29951013 | 29901418 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 22 | 29951017 | 29901245 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 23 | 29951022 | 29901245 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 24 | 29951027 | 29901288 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 25 | 29951037 | 29901288 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 26 | 29951042 | 29951032 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 27 | 29951047 | 29951032 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 28 | 29951053 | 29901424 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 29 | 29951056 | 29951032 | 2024-04-19 21:11:39.034639 | 30 | W | country => DE,city => Tangermünd… |
| 30 | 29951602 | 29907027 | 2024-04-19 21:11:39.076664 | 30 | W | <null> |
| 31 | 29950476 | 29902534 | 2024-04-19 21:11:39.002658 | 30 | N | housenumber => 48,street => Frie… |
| 32 | 102416 | 48244940 | 2024-04-20 03:29:57.612966 | 0 | W | <null> |
| 33 | 29950481 | 29900820 | 2024-04-19 21:11:39.002658 | 30 | W | <null> |
| 34 | 29950485 | 29905922 | 2024-04-19 21:11:39.002658 | 30 | N | housenumber => 29,street => "Ste… |

*Obrázek 12: tabulka placex - 48 mil. záznamů pro sousední země ČR. Zdroj: Vlastní zpracování*

### 6.3.3 Hierarchie dat

Adresní rank(viz obrázek č. 11) v databázi Nominatimu hraje zásadní roli v určení, jak se geografické objekty zobrazí v hierarchii adres. Rank je číselný ukazatel, který přiděluje každému místu v tabulce placex jeho relevanci a úroveň v rámci adresního systému. Tento systém ranků umožňuje Nominatimu rozhodnout, jaká místa by měla být zahrnuta v adresních dotazech, a definuje logiku spojení mezi různými typy geografických údajů.

Ranky 1 až 3 jsou obvykle nevyužité a místa s těmito ranky se v adresních hierarchiích neobjevují. Rank 4 představuje zemi, která je obvykle odvozena nepřímo z kódu země, a neobjevuje se přímo ve struktuře adres objektu. Ranky 5 až 9 představují stát, 10 až 12 kraj, 13 až 16 město, 17 až 21 předměstí, 22 až 24 čtvrť a rank 25 označuje specifické objekty jako náměstí, farmy a lokalit. Ulice jsou reprezentovány ranky 26 a 27, zatímco ranky 28 až 30 představují bod zájmu (POI) nebo čísla domů.

Zvláštností je rank 25, který může být nadřazeným bodem pro objekty jako náměstí nebo budovy, ale nemůže být nadřazeným bodem pro ulice. Tento rank se používá pro místní

funkce, které jsou technicky na stejné úrovni jako ulice nebo pro místa, která by obvykle neměla být zobrazena v adresách, pokud nejsou explicitně označena (např. pomocí tagu place=locality).

Díky této hierarchii ranků může Nominatim efektivně zpracovat a uspořádat adresní data, což zajišťuje přesné a relevantní výsledky při geokódování a reverzním geokódování. Ranky adres tak slouží jako nástroj pro rozlišení a organizaci geografických dat, umožňující uživatelům Nominatimu a dalším systémům rozpoznat strukturu a vztahy mezi různými typy míst.

### 6.3.4 Adresy

V kontextu Nominatimu představují adresy nejdůležitější prvek pro naší cílovou aplikaci a jsou tedy centrálním bodem našeho zájmu. Každý záznam adresy může obsahovat různé atributy, jako jsou housenumber (číslo domu), street (ulice), city (město), country_code (kód země) a postcode (PSČ), což usnadňuje jejich identifikaci a propojení s dalšími geografickými entitami. Například záznamy s vyplněnými poli housenumber a street představují konkrétní adresní místa, zatímco záznamy pouze s city a country ukazují na vyšší úroveň v adresní hierarchii.

Vizualizace dat v Nominatimu dále umožňuje snadné rozpoznání vzorců a hierarchií. Například, záznamy s hodnotou <null> v určitém sloupci naznačují absenci specifického informačního elementu, což může být užitečné při analýze neúplných dat nebo při definování dotazů pro získání dat s požadovanou úrovní specifičnosti.

Jak již bylo zmíněno v kapitole o hierarchi 6.3.3, typy záznamů se rozlišují především podle atributu rank address. tudíž kdybych například narazili na záznam s typem rank address 16(obec), šli přes jeho place_id a hledali záznamy toto place_id je rovno parent_place_id, objevili bychom jeho podřezené entity, ať už části měst či ulice nebo dokonce rovnou adresní body (pokud by tato obec představovala vesnici).

## 6.4 Shrnutí

V této kapitole jsme se věnovali komplexnímu procesu integrace a využití nástroje Nominatim pro transformaci a analýzu adresních dat z OpenStreetMap (OSM) do platformy Grasshopper. Nominatim jako volně dostupný geokódovací nástroj poskytl efektivní způsob práce s geoprostorovými daty, který byl adaptován pro potřeby této diplomové práce prostřednictvím Dockeru a specifických skriptů.

Implementace Nominatimu byla realizována s ohledem na efektivitu zpracování dat a snadnou integraci do existující infrastruktury. Důraz byl kladen na schopnost nástroje zpracovat více regionů současně a aktualizovat je v pravidelných intervalech, což bylo dosaženo prostřednictvím vlastního Docker image a příslušných konfiguračních skriptů.

V rámci této práce byly nastaveny a úspěšně otestovány procesy pro import dat do Nominatimu, indexaci a vzájemné provazování entit. Byla vytvořena a demonstrována struktura databáze schopná reprezentovat adresní data s různými úrovněmi detailů, a to včetně provázání entit skrze atributy place_id a parent_place_id.

Diskutovali jsme o různých aspektech datového modelu Nominatimu, včetně hierarchie adresních dat a jejich reprezentace v databázové tabulce placex. Byla zdůrazněna flexibilita a rozšiřitelnost tohoto přístupu, který umožňuje uživatelům efektivní dotazování a vizualizaci komplexních prostorových vztahů.

Závěrem lze konstatovat, že využití nástroje Nominatim v rámci této diplomové práce přineslo nejen akademické poznatky v oblasti geoprostorových dat, ale také hodnotné zkušenosti s implementací reálného datového projektu. Práce s Nominatimem tak položila základy pro budoucí rozvoj a aplikace v oblasti adresních a prostorových dat. Výsledky této kapitoly poskytují pevný základ pro další rozšíření a využití adresních dat.

# 7 Transformace do cílového datového modelu

Cílem kapitoly 7, zaměřené na transformaci dat, je převést geoprostorová data z formátu, ve kterém jsou ukládána a spravována v systému Nominatim, do cílového formátu aplikace Grasshopper. To zahrnuje procesy extrakce, transformace a nahrání (ETL), které zajišťují převod datových struktur, atributů a vztahů tak, aby odpovídaly modelu a potřebám aplikace Grasshopper.

V kontextu Nominatimu se data nacházejí ve formě tabulek s komplexními vazbami a hierarchiemi. Tyto informace musí být zredukované a transformované do jednoduššího a přímočařejšího formátu, který vyžaduje aplikace Grasshopper. Tento formát by měl umožňovat snadný přístup k datům, jednoduché dotazování a efektivní interakci s uživatelským rozhraním aplikace.

Transformace zahrnuje identifikaci a mapování klíčových prvků jako jsou ulice, adresní body, města, části měst, pošty a regiony, a jejich přiřazení do odpovídajících tříd a objektů, které aplikace Grasshopper používá. Součástí tohoto procesu je také normalizace a sjednocení různých formátů a konvencí, například sjednocení kódování textu, formátování poštovních kódů a překlad jmen a dalších atributů.

Pro účely této transformace bude vyvinut a implementována v jazyce Java, které budou sloužit k extrakci dat z Nominatimu, jejich transformaci do požadované struktury a nahrání do databáze Grasshopper. Důležité bude také zajistit, že všechny transformace zachovávají integritu dat, nepřidávají redundanci a zachovávají všechny potřebné vztahy mezi daty, aby byla zajištěna jejich přesnost.

Konečným výstupem tohoto procesu bude databáze schopná poskytnout uživatelům aplikace Grasshopper rychlé a spolehlivé informace, což zvýší uživatelskou spokojenost a rozšíří možnosti použití aplikace v reálném světě.

## 7.1 Struktura aplikace

Aplikace Grasshopper, cíl transformace dat z Nominatimu, je strukturována s důrazem na modulárnost, efektivní správu dat a uživatelskou přívětivost. Struktura aplikace je navržena tak, aby podporovala různé úrovně abstrakce dat a zpracování, což zajišťuje její rozšiřitelnost a schopnost integrace s širokou škálou geografických datových zdrojů.

Kernem aplikace je robustní databázový model, který je navržen pro ukládání a dotazování geografických dat. Tento model zahrnuje objekty jako jsou Street, AddressPoint,

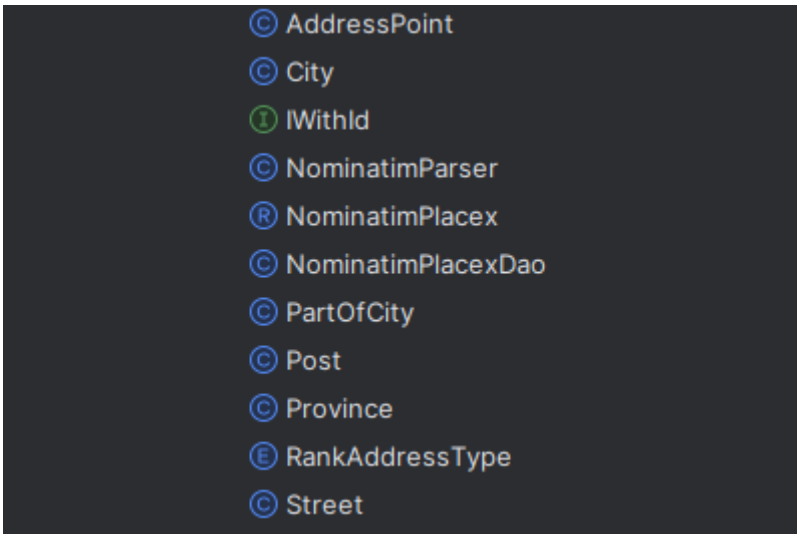
City, PartOfCity, Post, a Province, které odpovídají různým entitám geografických informací. Každý z těchto objektů je implementován jako třída v Java, což umožňuje aplikaci spravovat a manipulovat s daty v objektově orientovaném přístupu.

Rozhraní aplikace je navrženo tak, aby bylo intuitivní a přizpůsobivé různým typům uživatelů. Uživatelské rozhraní umožňuje interakci s daty prostřednictvím jednoduchých dotazů nebo složitějších analýz, zatímco API poskytuje bránu pro externí systémy a služby, které mohou využívat zpracovaná data.

Proces transformace dat zahrnuje nástroj NominatimParser, který je zodpovědný za čtení a transformaci dat získaných z Nominatimu a jejich převod do struktury, kterou Grasshopper požaduje. NominatimPlacexDao pak slouží jako most mezi surovými daty a aplikací, zajišťuje dotazování a extrakci potřebných informací z databáze Nominatim.

Součástí aplikace jsou také specifické třídy pro zpracování a prezentaci dat, jako RankAddressType, které mapují složitost adresních rankech do jasně definovaných typů adres v aplikaci. Tyto třídy pomáhají zachovat logiku hierarchie a vztahů mezi různými adresními údaji.

Výsledkem je architektura, která je schopna plynule přijímat, zpracovávat a prezentovat geografická data, čímž umožňuje aplikaci Grasshopper nabízet uživatelům bohaté a dynamické geografické informace pro různé použití od jednoduchého vyhledávání až po složité analýzy.




*Obrázek 13: struktura aplikace, její třídy. Zdroj: Vlastní zpracování*

### 7.1.1 Nastavení tříd

Při nastavení tříd v aplikaci Grasshopper je zásadní zajistit, aby každá třída odpovídala konkrétní entitě geografických dat a zároveň byla kompatibilní s datovým modelem zdroje, v tomto případě Nominatimu. Třídy jako Street, AddressPoint, City, PartOfCity, Post, a Province jsou definovány s cílem odrazit strukturu a vztahy dané v geografických datech.

Každá třída je vybavena souborem atributů, které popisují vlastnosti dané geografické entity - například, pro ulice to může být název, délka, spojené adresy, a případně její geografické souřadnice. Metody přidružené k těmto třídám umožňují manipulaci s objekty, jako jsou vytváření, čtení, aktualizace a mazání (CRUD operace), a zahrnují logiku potřebnou pro spojování entit, vyhledávání v databázi a vykonávání specifických dotazů.

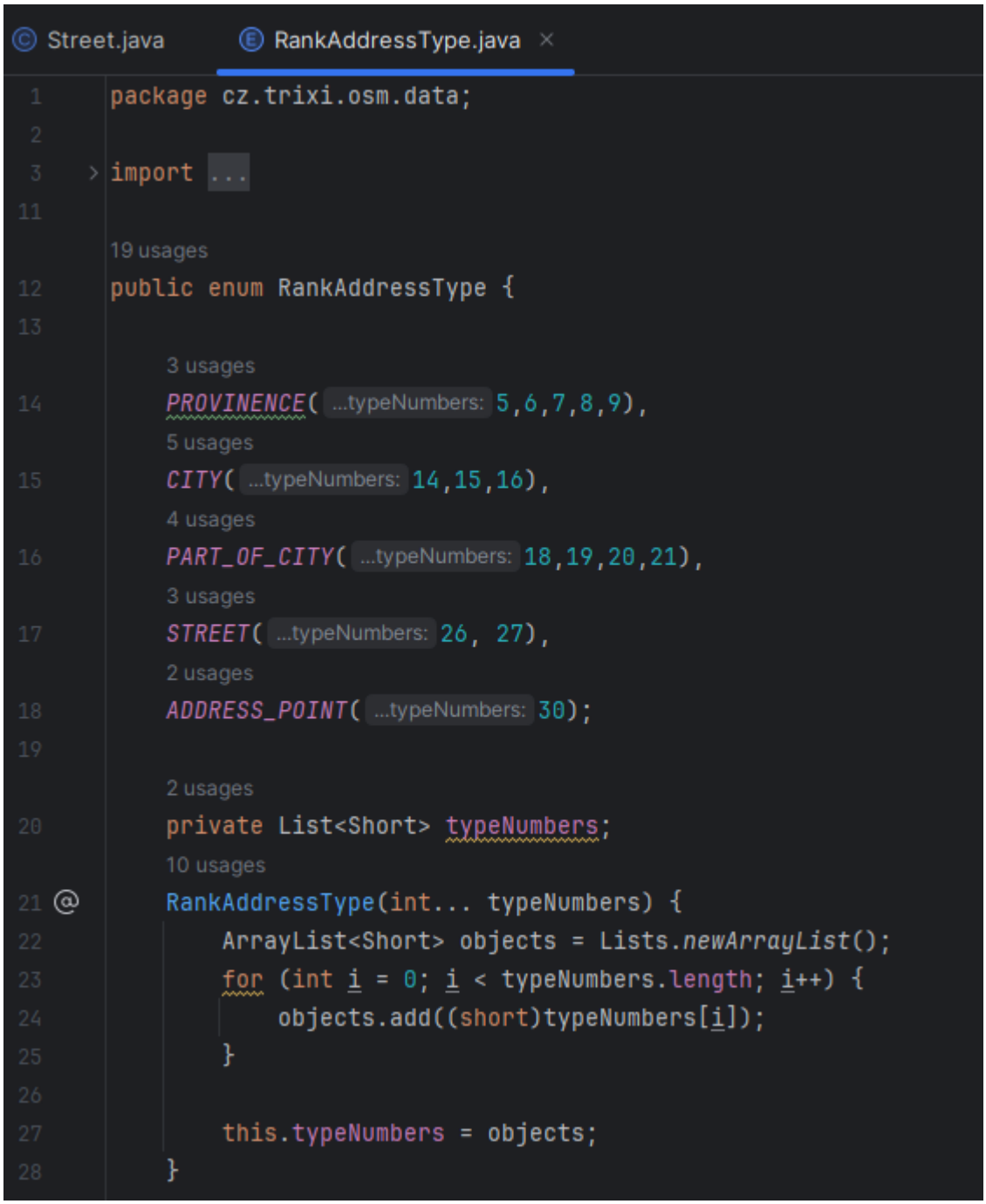
### 7.1.2 Rozpoznání typu entity

Aby bylo možné efektivně rozlišit mezi různými geografickými entitami, jako jsou ulice, města, části měst a další, aplikace využívá koncept enum RankAddress, který je odvozen od adresních ranků Nominatimu.

Enum RankAddress představuje sadu konstant, které mapují hodnoty adresních ranků Nominatimu na konkrétní typy entit v aplikaci. To umožňuje aplikaci jednoznačně určit, s jakým typem entity pracuje, což je nezbytné pro správnou funkci dotazů, vyhledávání a dalších operací spojených s geodaty. Tento přístup zajišťuje, že každý geografický objekt je řádně kategorizován a zpracován v souladu s jeho významem v adresní hierarchii.

Při zpracování dat aplikace načítá place_id z Nominatimu a pomocí RankAddress určuje, zda se jedná o ulici, město, nebo jiný typ entity. Tímto způsobem může aplikace dynamicky přizpůsobit svůj dotaz nebo analytické operace, aby reflektovaly specifika každé entity. Například, pokud RankAddress indikuje, že se jedná o ulici, aplikace může vyvolat specifické metody nebo operace určené pro práci s ulicemi.

Výhodou použití enum RankAddress je také snížení možnosti chyb při programování, jelikož vývojáři mohou využít výhod staticky typovaného jazyka, kde každá konstanta v enumu má pevně definovanou hodnotu, což usnadňuje porozumění kódu a jeho údržbu.




*Obrázek 14: Rank Address Enum. Zdroj: Vlastní zpracování*

## 7.2 Parsování dat

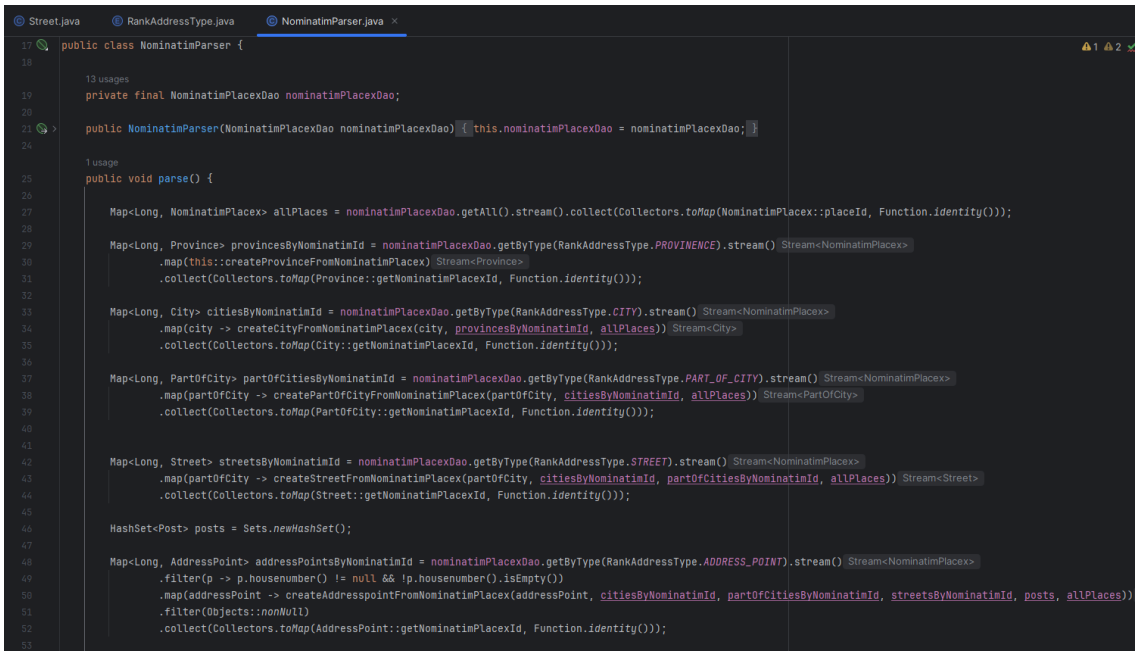
Třída NominatimParser je klíčovou součástí aplikace Grasshopper a je zodpovědná za zpracování a transformaci surových dat z Nominatimu do strukturované formy požadované cílovou aplikací. Parsování dat představuje proces extrakce relevantních informací z komplexních datových sad a jejich převedení do přesně definovaných objektů a tříd, jako jsou Street, AddressPoint, City, a další.

NominatimParser funguje tak, že načte datové sady a identifikuje klíčové atributy potřebné pro každou entitu. Využívá informace o adresních rankech a prostřednictvím mapování vytváří instance objektů odpovídající různým typům adresních prvků. Tento proces zahrnuje:

- Čtení hodnot z tabulky placex, jako jsou place_id, parent_place_id, názvy ulic, čísla domů, PSČ, a další.
- Rozpoznání typu entity pomocí enumu RankAddress a určení, jaké objekty a vztahy je třeba vytvořit.
- Extrahování a normalizace geografických souřadnic a jiných prostorových informací.
- Transformaci těchto informací do formátu, který je přímo použitelný a uchovatelný v aplikaci Grasshopper.

Jedná se o náročný proces, který vyžaduje přesné a promyšlené algoritmy k zajištění integrity a správné interpretace dat. Parsování musí být dostatečně robustní, aby zvládlo různé anomálie a nekonzistence, které se v surových datech mohou vyskytovat.

NominatimParser je vybaven sadou metod pro zpracování a validaci dat, což zahrnuje ověření správnosti formátů a vyhledání možných chyb v datech. Tato funkcionalita je zásadní pro udržení kvality dat v aplikaci a pro poskytování přesných a spolehlivých výsledků uživatelům.




*Obrázek 15: Třída NominatimParser odpovědná za parsování tabulky placex.Zdroj: Vlastní zpracování*

### 7.2.1 Vytváření objektů

Významnou částí transformačního procesu v aplikaci Grasshopper je vytváření instancí objektů, zejména objektu AddressPoint, který reprezentuje jednotlivé adresní body, toto je zcela nejdůležitější třída pro následnou práci s adresami v platforme Grasshopper.

Vytváření objektu AddressPoint začíná načtením a analýzou příslušných dat získaných z Nominatimu. Poté co jsme rozparsovali databázi s daty v datovém formátu nástroje Nominatimu, je třeba vytvořit instance pro tato data a správně přiřadit dané attributy z databáze instancím této třídy. Můžeme to rozdělit do několika kroků:

- Extrahování nezbytných atributů z datové sady, včetně place_id, geografických souřadnic, housenumber, street, postcode, a případných dalších metadata.
- Inicializaci nové instance třídy AddressPoint s použitím těchto dat. To zahrnuje přiřazení hodnot k atributům a zajištění, že každý AddressPoint má unikátní identifikátor.
- Aplikaci validací, které zajistí, že data jsou kompletní a bez chyb, a pokud jsou nalezeny nesrovnalosti, jsou správně ošetřeny.
- Vytvoření vztahů s ostatními objekty, jako jsou Street nebo City, aby bylo zajištěno správné zařazení adresního bodu do geografického kontextu.

Proces vytváření objektů AddressPoint je navržen s důrazem na přesnost, efektivitu a udržitelnost.

### 7.2.2 Propojení mezi entitami

Podobně jako v Nominatimu, kde jsou entity propojeny pomocí place_id a parent_place_id, aplikace Grasshopper implementuje sofistikovaný systém propojení, který využívá enum RankAddressType k určení rodičovských entit.

```
private NominatimPlacex getParentByType(NominatimPlacex place, RankAddressType placeType, Map<Long, NominatimPlacex> allPlaces) {
    if (place == null) {
        return null;
    }
    if (placeType.getTypeNumbers().contains(place.rankAddress())) {
        return place;
    }
    NominatimPlacex parent = allPlaces.get(place.parentPlaceId());
    while (parent != null) {
        if (placeType.getTypeNumbers().contains(parent.rankAddress())) {
            return parent;
        }
        parent = allPlaces.get(parent.parentPlaceId());
    }
    return null;
}
```

*Obrázek 16: Funkce pro nalezení nadřazené entity. Zdroj: Vlastní zpracování*

Pro identifikaci a propojení s rodičovskými entitami je v aplikaci Grasshopper používána funkce getParentByType, která efektivně vyhledává v hierarchii entity podle specifikovaného typu. Tato metoda přijímá tři parametry: objekt place (aktuální entita), placeType (typ entity, který určuje hledaný rodičovský typ), a allPlaces (mapa všech entit, kde klíčem je place_id a hodnotou objekt entity).

**Funkce pracuje následovně**:

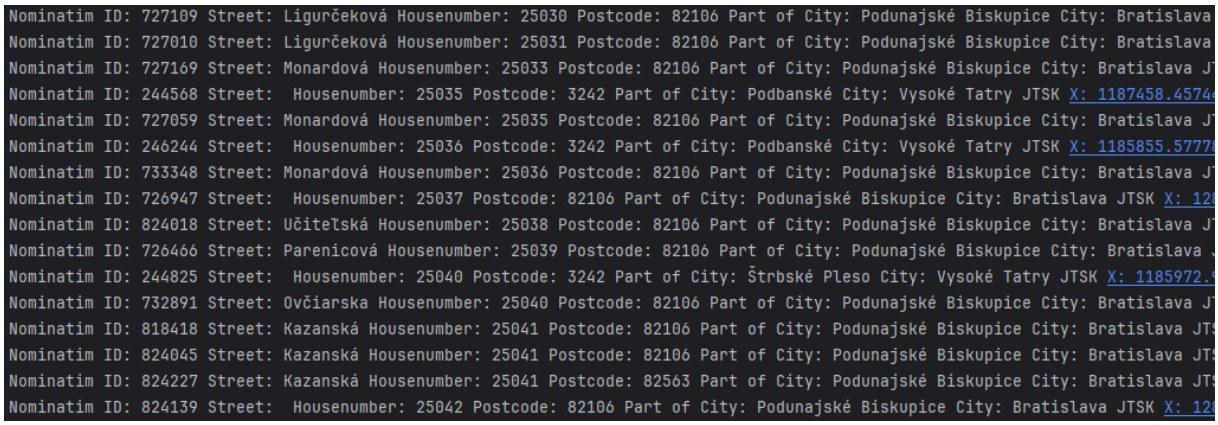
Počáteční kontrola: Nejprve zkontroluje, zda předaná entita place není null. Pokud je null, vrátí null jako výsledek, jelikož neexistuje žádná entita, kterou by bylo možné zpracovat.

Ověření ranku: Následně funkce zkontroluje, zda rank aktuální entity odpovídá ranku hledaného typu, definovaného v placeType. Pokud ano, vrátí tuto entitu jako rodiče.

Procházení hierarchií: Pokud aktuální entita neodpovídá hledanému typu, funkce přistoupí k jejímu rodiči pomocí parentPlaceId a tento proces opakuje. Procházení hierarchie pokračuje, dokud není nalezen odpovídající rodič, nebo dokud nejsou prohledány všechny rodičovské entity.

Vrácení výsledku: Jakmile je nalezena entita s odpovídajícím rankem, je vrácena jako výsledek. Pokud taková entita není nalezena, funkce vrátí null.

Tato metoda umožňuje aplikaci dynamicky rekonstruovat adresní hierarchii a přiřadit každé adrese nebo geografické entitě její správný rodičovský prvek podle specifikovaného kritéria, což je nezbytné pro efektivní organizaci a vyhledávání geografických dat.




*Obrázek 17: Výpis z konzole. Zdroj: Vlastní zpracování*

Zde v obrázku lze vidět výpis daných instancí třídy adresní bod a jeho atributy jako Street (ulice), City (město), Housenumber (Číslo popisné) a další.

### 7.2.3 Vytvoření tabulek a nahrání instancí tříd do databáze

Pro správné fungování aplikace je nezbytné zajistit, že všechny potřebné tabulky v databázi jsou správně definovány a naplněny daty. Proces vytváření tabulek a nahrávání datových instancí zahrnuje několik kritických kroků, které zajišťují, že aplikace může efektivně spravovat a vyhledávat geografické informace.

**Definice schématu databáze:**

Před zahájením jakékoliv operace s daty je nezbytné definovat a implementovat databázové schéma, které přesně odráží strukturu a požadavky aplikace. Tento krok zahrnuje vytvoření tabulek, jako jsou sr_kraj, sr_okres, sr_obec, sr_cast_obce, sr_ulice, sr_posta, a sr_adresni_misto, s přesně specifikovanými sloupci pro každou entitu. Správná definice schématu zahrnuje také nastavení klíčů a indexů, které zlepšují výkon a usnadňují integritu dat při dotazování.

**Implementace SQL operací:**

Využití JdbcTemplate pro dávkové operace je efektivní strategie pro minimalizaci zatížení databázového serveru a zrychlení procesu vkládání dat. Tato technika umožňuje aplikaci zpracovávat velké objemy dat s menším počtem připojení k databázi, což snižuje overhead spojený s otevíráním a zavíráním databázových transakcí. Efektivní implementace SQL operací zahrnuje také správné plánování transakcí a vyvažování zatížení, aby byly operace prováděny rychle a bez chyb.

**Generování unikátních identifikátorů:**

Správné přiřazení unikátních identifikátorů všem instancím před jejich vložením do databáze je klíčové pro udržení referenční integrity. Funkce generateIds() zajišťuje, že každý objekt má jedinečné ID, což je nezbytné pro odstranění jakýchkoli duplikátů a konfliktů v databázi. Tento krok je zásadní pro sledování a správu objektů v systému, což umožňuje snadnější údržbu a manipulaci s daty.

**Nahrávání dat:**

Metody, které jsou speciálně vyvinuty pro vkládání dat do jednotlivých tabulek, musí být navrženy tak, aby efektivně zvládaly nahrávání dat s ohledem na specifika dané entity. Tyto metody nejenže zohledňují atributy entit, ale také řeší vazby a závislosti mezi různými entitami, což umožňuje zachování struktury a hierarchie v databázi. Například, přiřazení správného obecního ID ulici zajišťuje, že všechny adresy v ulici jsou korektně spojeny s odpovídající obcí.

**Zajištění integrity dat:**

Po nahrání dat do databáze je nezbytné provést kontrolu integrity, aby bylo zajištěno, že všechna data jsou správně reprezentována a není zde žádná nekonzistence. To zahrnuje validaci datových typů, ověření, že všechna data splňují definované omezení, a zajištění, že všechny relace mezi tabulkami jsou správně nastaveny. Tento krok je klíčový pro zajištění spolehlivosti a přesnosti datových dotazů, které systém provádí.

Tento proces je nezbytný pro udržení kvality a přesnosti dat uložených v databázi a zajišťuje, že aplikace může efektivně reagovat na dotazy uživatelů a poskytovat spolehlivé geografické informace.

### 7.2.4 Insert adresního místa

Vkládání je prováděno pomocí metody třídy NominatimPlacexDao:

```
public void insertAdresniMisto(Collection<AddressPoint> addressPoints) {
    addressPoints.removeIf(addressPoint -> addressPoint.getCity().getId() == 0);
    generateIds(addressPoints);
    jdbcTemplate.batchUpdate(
            sql: "INSERT INTO sr_adresni_misto (id, popisne_cislo, orientacni_pismeno, psc, cast_obce_id, " +
                    "ulice_id, city_id, latitude, longitude, x, y) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)",
            addressPoints,
            batchSize: 100,
            (ps, addressPoint) -> {
                ps.setInt( parameterIndex: 1, addressPoint.getId());
                ps.setInt( parameterIndex: 2, addressPoint.getHousenumber());
                ps.setString( parameterIndex: 3, addressPoint.getHousenumberOrinetChar());

                ps.setInt( parameterIndex: 4, addressPoint.getPsc());
                if (addressPoint.getPartOfCity() == null)
                    ps.setObject( parameterIndex: 5,  x: null);
                else
                    ps.setInt( parameterIndex: 5, addressPoint.getPartOfCity().getId());
                if (addressPoint.getStreet() == null)
                    ps.setObject( parameterIndex: 6,  x: null);
                else
                    ps.setInt( parameterIndex: 6, addressPoint.getStreet().getId());
                ps.setInt( parameterIndex: 7, addressPoint.getCity().getId());
                ps.setDouble( parameterIndex: 8, addressPoint.getLatitude());
                ps.setDouble( parameterIndex: 9, addressPoint.getLongtitude());
                ps.setDouble( parameterIndex: 10, addressPoint.getJtskX());
                ps.setDouble( parameterIndex: 11, addressPoint.getJtskY());
            });
}
```

*Obrázek 18: Metoda InsertAdresniMisto. Zdroj: Vlastní zpracování*

Filtrace nevalidních dat: Před zpracováním vlastního vkládání dat metoda odstraňuje všechny adresní body, které nemají validní id města (tj. addressPoint.getCity.getId() == 0). Toto zajistí, že do databáze budou vložena pouze relevantní a kompletní data.

Generování identifikátorů: Každému adresnímu bodu je přiřazen unikátní identifikátor pomocí funkce generateIds. Tento krok je nezbytný pro zachování jednoznačné identifikace záznamů v databázi a pro zajištění integrity dat.

Dávkové vkládání dat: Metoda využívá jdbcTemplate.batchUpdate pro hromadné vkládání dat, což zefektivňuje proces tím, že omezuje počet individuálních transakcí provedených databází. Dávkové zpracování také snižuje zatížení databáze a zvyšuje celkovou rychlost vkládání dat.

Nastavení parametrů: Pro každý adresní bod jsou nastaveny parametry SQL dotazu, které zahrnují identifikátory, domovní čísla, orientační čísla, poštovní směrovací čísla, a

identifikátory pro část města, ulici a město, stejně jako geografické souřadnice. Speciální zpracování pro null hodnoty (pokud ulice nebo část města nejsou definovány) zajišťuje, že databáze bude obsahovat konzistentní a úplné informace.

Zpracování geografických souřadnic: Kromě základních informací jsou do databáze vloženy také souřadnice latitude, longitude a transformované souřadnice JTSK (x, y), což umožňuje další geoprostorové analýzy a aplikace.

## 7.3 Výsledky nahrávání dat

Po úspěšném provedení procesu nahrávání dat do databáze je nyní možné vizuálně prozkoumat výsledky, jak jsou zaznamenány v tabulce sr_adresni_misto. Následující obrázek ukazuje výřez tabulky s konkrétními záznamy, které reprezentují adresní místa.

| id | popisne_cislo | or... | psc | cast_obce_id | ulice_id | city_id | latitude | longitude |
|---|---|---|---|---|---|---|---|---|
| 125 | 241 | <null> | 93574 | 3378 | 141904 | 754 | 47.966606999999996 | 18.762713725762936 |
| 126 | 265 | <null> | 93574 | <null> | <null> | 754 | 47.96398045 | 18.758442556067124 |
| 127 | 3 | <null> | 93574 | 3378 | 142465 | 754 | 47.9651864 | 18.763598923158824 |
| 128 | 263 | <null> | 93574 | <null> | <null> | 754 | 47.96376175 | 18.759087829580515 |
| 129 | 269 | <null> | 93574 | <null> | <null> | 754 | 47.9705979 | 18.75230798804403 |
| 130 | 53 | <null> | 93574 | 3378 | 67454 | 754 | 47.96950635 | 18.753495809 |
| 131 | 131 | <null> | 93574 | 3378 | 136302 | 754 | 47.96999945 | 18.75230257416431 |
| 132 | 2 | <null> | 93574 | 3378 | 67454 | 754 | 47.9696588 | 18.75302111027292 |
| 133 | 49 | <null> | 93574 | 3378 | 141888 | 754 | 47.969088600000006 | 18.752852147972902 |
| 134 | 50 | <null> | 93574 | 3378 | 141888 | 754 | 47.9691063 | 18.752742932503736 |
| 135 | 84 | <null> | 93574 | 3378 | 141888 | 754 | 47.9691975 | 18.752355943895967 |
| 136 | 56 | <null> | 93574 | 3378 | 67536 | 754 | 47.9688276 | 18.752402636360294 |
| 137 | 14 | <null> | 93574 | 3378 | 142757 | 754 | 47.97161455 | 18.75029346483051 |
| 138 | 1 | <null> | 93574 | 3378 | 141931 | 754 | 47.96886115 | 18.751511393033336 |
| 139 | 19 | <null> | 93574 | 3378 | 67474 | 754 | 47.97170534999999 | 18.75337260931531 |
| 140 | 136 | <null> | 93574 | 3378 | 67474 | 754 | 47.9715709 | 18.753339038193403 |
| 141 | 22 | <null> | 93574 | 3378 | 67474 | 754 | 47.9718803 | 18.75381900866921 |
| 142 | 201 | <null> | 94361 | <null> | <null> | 4176 | 47.9291798 | 18.750657444755365 |
| 143 | 121 | <null> | 93574 | 3378 | 142757 | 754 | 47.97205865 | 18.750382175314996 |
| 144 | 140 | <null> | 93574 | 3378 | 67474 | 754 | 47.97212015 | 18.753359053499402 |
| 145 | 119 | <null> | 93574 | 3378 | 142757 | 754 | 47.97243265 | 18.750374119016957 |

*Obrázek 19: tabulka sr_adresni_misto. Zdroj: Vlastní zpracování*

V tabulce je patrné, že každý záznam má přiděleno unikátní ID, což je klíčové pro jednoznačnou identifikaci a další manipulaci s daty. Sloupce popisne_cislo a psc (poštovní směrovací číslo) poskytují specifické informace týkající se adresního bodu. Sloupce cast_obce_id, ulice_id, a city_id jsou využívány k určení vztahu mezi adresním místem a ostatními geografickými entitami, jako jsou části obcí, ulice a města. To umožňuje efektivní spojování a vyhledávání informací na základě geografického umístění.

Dále sloupce latitude a longitude obsahují geografické souřadnice každého adresního bodu, což je základ pro geolokační služby a mapové aplikace. Souřadnice x a y v koordinátech JTSK (Jednotný trigonometrický katastrální sít) jsou klíčové pro integraci s českými geodetickými a katastrálními systémy.

Přítomnost hodnot <null> v některých záznamech naznačuje, že pro daný adresní bod nebyly specifikovány všechny informace, například příslušnost k ulici. Tyto případy nastávají v situaci, kdy se například daný adresní bod nachází ve vesnici, to znamená že v Nominatim struktuře dat jeho další nadřazená entita je až city - obec. Tudíž zde jeho ulice a část obce nejsou vyplněny, což ale pro funkčnost platformy Grasshopper není problém.

Výsledky v této tabulce jsou klíčovým základem pro adresní systém aplikace a umožňují provádění různorodých dotazů, od jednoduchého vyhledání adresy až po složité geoprostorové analýzy.

# 8 Diskuse

Tato kapitola se zaměřuje na kritickou analýzu zvolených nástrojů a metodik použitých v rámci diplomové práce. Důležitost této diskuse spočívá v objasnění, jak výběr technologií a přístupů ovlivňuje výsledky projektu a jaké jsou potenciální omezení a výhody spojené s implementací. Diskuse nejenže poskytuje hlubší porozumění, proč byly určité technologie jako Nominatim a osm2pgsql vybrány, ale také reflektuje na jejich praktické dopady v reálných aplikacích. Analýza těchto aspektů je klíčová pro pochopení, jak teoretické koncepty fungují ve skutečném světě, a jak mohou být použity nebo modifikovány pro budoucí projekty a výzkumy.

## 8.1 Výhody a nevýhody zvoleného řešení

Zásadním přínosem zvoleného řešení využívajícího Nominatim a osm2pgsql pro správu a analýzu geografických dat je jejich otevřenost a široká podpora. Díky otevřenému zdroji jsou tyto nástroje dostupné bez licenčních poplatků, což snižuje náklady na projekt a umožňuje snadné škálování aplikace bez dodatečných finančních závazků. Tato flexibilita je klíčová pro start-upy a výzkumné týmy s omezeným rozpočtem. Navíc, široká uživatelská a vývojářská komunita zajišťuje pravidelné aktualizace a robustní dokumentaci, což usnadňuje řešení problémů a implementaci nových funkcí.

### Nevýhody a omezení

Přestože jsou otevřené nástroje jako Nominatim a osm2pgsql extrémně užitečné, nesou s sebou i určitá omezení. Významným omezením je závislost na kvalitě a aktualitě otevřených datových zdrojů, jako je OpenStreetMap. Data OSM jsou sice rozsáhlá a pravidelně aktualizovaná, ale mohou obsahovat nepřesnosti nebo neúplnosti způsobené jejich crowdsourced povahou. Toto může vést k výzvám ve scénářích, kde jsou vyžadována data s vysokou mírou přesnosti, například v navigačních a záchranných aplikacích. Další nevýhodou je technická složitost spojená s nastavením a správou vlastní serverové infrastruktury pro běh těchto nástrojů, což může být pro nové uživatele náročné.

### Reflexe nad zvoleným řešením

Při hodnocení výhod a nevýhod je důležité zvážit kontext aplikace, ve které se řešení používá. V případě tohoto projektu převažují výhody dostupnosti a škálovatelnosti otevřených nástrojů nad potenciálními problémy s přesností dat a technickými požadavky. Pro budoucí rozvoj je však klíčové pokračovat ve vývoji a přizpůsobování nástrojů tak, aby co nejlépe

vyhovovaly specifickým potřebám uživatelů a zároveň zůstaly udržitelné a efektivní. Možné směry rozvoje zahrnují integraci s dalšími datovými zdroji pro zvýšení spolehlivosti, stejně jako rozvoj uživatelsky přívětivějších nástrojů pro správu a analýzu dat.

# Závěr

Tato diplomová práce "Analýza adresních dat z OpenStreetMap a jejich automatizovaná transformace do platformy Grasshopper " byla zaměřena na vývoj a implementaci systému pro efektivní transformaci geoprostorových dat z modelu OpenStreetMap do struktury vhodné pro využití v aplikaci Grasshopper. Hlavním cílem bylo nejen demonstrovat technickou proveditelnost takové transformace, ale také poskytnout ucelené řešení, které může být aplikováno v reálných projektech a výzkumech v oblasti geografických informačních systémů.

Práce úspěšně dosáhla stanovených cílů prostřednictvím vývoje nástrojů a metodik, které byly podrobně testovány a optimalizovány pro zpracování a analýzu geodat. Během projektu bylo provedeno několik klíčových úkonů, včetně výběru vhodného technologického stacku, navrhování databázového schématu a implementace softwarových modulů, které zajišťují transformaci dat s vysokou přesností a efektivitou.

Analýza a diskuse ukázaly, že ačkoliv existují určitá technická a metodologická omezení spojená s použitím otevřených dat a nástrojů, přínosy z hlediska nákladové efektivity, adaptability a rozšířené podpory komunity tato omezení převažují. Tato práce tak přispívá k rozvoji znalostní báze v GIS oblasti a nabízí solidní základ pro další výzkum a rozvoj v této dynamicky se rozvíjející oblasti.

Jako další kroky a potenciální směry budoucích prací se nabízí rozšíření funkcionality vytvořeného systému o další analytické nástroje, zlepšení uživatelské přívětivosti a automatizace, a dále prohloubení integrace s jinými platformami a datovými zdroji. Tyto aktivity mohou významně přispět k přesnosti, rychlosti a užitečnosti geografických informačních systémů v praxi.

V závěru lze konstatovat, že práce splnila své hlavní cíle a poskytla cenné poznatky a nástroje, které mohou být využity pro řešení komplexních výzev v oblasti zpracování a analýzy geoprostorových dat. Nad rámec akademického přínosu práce také ukazuje praktickou aplikovatelnost ve světě reálných aplikací, což zdůrazňuje její hodnotu a relevanci pro odbornou veřejnost.

# Použité zdroje


**Arcgis.** ArcGIS Maps SDK for Kotlin. *Developers ArcGIS.* [Online] [Citace: 22. Únor 2024.] https://developers.arcgis.com/kotlin/spatial-and-data-analysis/spatial-references/.

**Aws Amazon, authors. 2024.** What is a Data Warehouse? *aws.amazon.* [Online] 2024. [Citace: 13. Únor 2024.] https://aws.amazon.com/what-is/data-warehouse/.

**Barron, Christopher, Neis, Pascal a Zipf, Alexander. 2014.** *A Comprehensive Framework for Intrinsic OpenStreetMap Quality Analysis.* [Výzkumný článek] Heidelberg : Department of Geography, University of Heidelberg, 2014. 1467-9671.

**Berga, Mariana a Figuereido, Rute. 2023.** Kotlin vs Java: the 12 differences you should know. *Imaginary Cloud.* [Online] 14. Září 2023. [Citace: 2024. Únor 22.] https://www.imaginarycloud.com/blog/kotlin-vs-java/.

**Davidson, Scott. 2024.** Grasshopper. *Grasshopper.* [Online] 2024. [Citace: 26. Únor 2024.] https://www.grasshopper3d.com/.

**Ferrari, Luca a Pirozzi, Enrico. 2020.** *Learn PostgreSQL - Build and manage high-performance database solutions.* Birmingham : Packt Publishing, 2020. 978-1-83898-528-8.

**Foody, Giles, a další. 2017.** *Mapping and the Citizen Senzor.* London : Ubiquity Press Ltd, 2017. 978-1-911529-17-0.

**Geofabrik GmbH, Data/Maps Copyright 2018 a contributors, OpenStreetMap. 2018.** Geofabrik. *Geofabrik.* [Online] 2018. [Citace: 2024. 4 12.] https://download.geofabrik.de/europe.html.

**GIS Geography. 2023.** GIS Geography. *https://gisgeography.com/data-engineering-gis/.* [Online] 2023. https://gisgeography.com/data-engineering-gis/.

**GISGeography. 2023.** 15 Python Libraries for GIS and Mapping. *GISGeography.* [Online] 29. Říjen 2023. [Citace: 21. Únor 2024.] https://gisgeography.com/python-libraries-gis-mapping/.

**Github wiki. 2016.** Geotools wiki. *Geotools wiki.* [Online] 1. Prosinec 2016. [Citace: 17. Únor 2024.] https://github.com/geotools/geotools/wiki/.

**Haklay, Mordechai (Muki) a Weber, Patrick. 2008.** *OpenStreetMap: User-Generated Street Maps.* [Dokument] London : University College London, 2008. 1558-2590.

**Jedlička, Karel. 2022.** *Prostorová data v analogové podobě* -. [Studijní článek] Plzeň : Západočeská univerzita v Plzni, 2022.

**Kimball, Ralph a Caserta, Joe. 2004.** *Data Warehouse ETL Toolkit - Practical Techniques for Extracting, Cleaning, Conforming and Delivering Data.* místo neznámé : John Wiley & Sons Inc, 2004. 0764567578ID.

**Locationtech Github GeoWave. 2022.** Locationtech Github GeoWave. *Locationtech Github GeoWave.* [Online] 17. 6 2022. [Citace: 21. Únor 2024.] https://locationtech.github.io/geowave/latest/overview.html.

**Locationtech Github JTS.** JTS Topology Suite. *JTS Topology Suite.* [Online] [Citace: 21. Únor 2024.] https://locationtech.github.io/jts/jts-features.html.

**Microsoft.** Extrakce, transformace a načítání (ETL). *Learn Microsoft.* [Online] [Citace: 27. Únor 2024.] https://learn.microsoft.com/cs-cz/azure/architecture/data-guide/relational-data/etl.

**Miller, Harvey J. a Han, Jiawei. 2009.** *Geographic Data Mining and Knowledge Discovery: Edition 2.* Boca Raton, Florida : CRC Press, 2009. 978-1420073973.

**Netrdová, Pavlína. 2010.** *Současné trendy v kvantitativní analýze geografických dat: možnosti a problémy prostorové analýzy dat.* Praha : Univerzita Karlova, Přírodovědecká fakulta, Katedra sociální geografie a region. rozvoje, 2010.

**Nominatim developer community. 2020.** Nominatim Documentation. *Nominatim 4.4.0 Manual.* [Online] 2020. [Citace: 2024. Duben 18.] https://nominatim.org/release-docs/4.4/develop/Database-Layout/.

**OpenStreetMap contributors, Wiki. 2024.** Nominatim. *Bibliographic details for Nominatim.* [Online] 25. Březen 2024. [Citace: 29. Duben 2024.] https://wiki.openstreetmap.org/w/index.php?title=Nominatim&oldid=2685718.

**—. 2021.** Osm2pgsql. *Bibliographic details for Osm2pgsql.* [Online] 4. Listopad 2021. [Citace: 2024. Duben 20.] https://wiki.openstreetmap.org/w/index.php?title=Osm2pgsql&oldid=2215447.

**OpenStreetMap Wiki, contributors. 2023.** Bibliographic details for History of OpenStreetMap. *Bibliographic details for History of OpenStreetMap.* [Online] 12. Listopad 2023. [Citace: 1. Únor 2024.] https://wiki.openstreetmap.org/w/index.php?title=History_of_OpenStreetMap&oldid=2615552.

**—. 2024.** Contributors. *OpenStreetMap Wiki.* [Online] 7. Únor 2024. [Citace: 12. Únor 2024.] https://wiki.openstreetmap.org/w/index.php?title=Contributors&oldid=2661333.

**—. 2020 .** Cs:Historie OpenStreetMap v České republice. *Cs:Historie OpenStreetMap v České republice.* [Online] 22 . Říjen 2020 . [Citace: 2024 . Únor 1.] https://wiki.openstreetmap.org/w/index.php?title=Cs:Historie_OpenStreetMap_v_%C4%8Cesk%C3%A9_republice&oldid=2052442.

**—. 2020.** Cs:Uzel. *OpenStreetMap Wiki.* [Online] 22. říjen 2020. [Citace: 1. Únor 2024.] https://wiki.openstreetmap.org/w/index.php?title=Cs:Uzel&oldid=2052506.

**—. 2023 .** Relation. *OpenStreetMap Wiki.* [Online] 14. Listopad 2023 . [Citace: 2. Únor 2024.] https://wiki.openstreetmap.org/w/index.php?title=Relation&oldid=2616506.

**—. 2024 .** Rendering. *OpenStreetMap Wiki.* [Online] 1. Únor 2024 . [Citace: 12. Únor 2024 .] https://wiki.openstreetmap.org/w/index.php?title=Rendering&oldid=2656877.

**—. 2023.** Tags. *OpenStreetMap Wiki.* [Online] 17. Červenec 2023. [Citace: 2. Únor 2024.] https://wiki.openstreetmap.org/w/index.php?title=Tags&oldid=2566368.

**—. 2024 .** Way. *OpenStreetMap Wiki.* [Online] 21. Leden 2024 . [Citace: 2. Únor 2024.] https://wiki.openstreetmap.org/w/index.php?title=Way&oldid=2650806.

**openstreetmap.cz. 2015.** https://openstreetmap.cz/. *https://openstreetmap.cz/.* [Online] 2015. [Citace: 1. Leden 2024.] https://openstreetmap.cz/.

**Paul A. Longley, Michael F. Goodchild, David J. Maguire, David W. Rhind. 2015.** *Geographic Information Systems and Science, 4th Edition.* Hoboken : Wiley, 2015. 978-1-118-67695-0.

**PECINOVSKÝ, Rudolf. 2022.** *Python - knihovny pro práci s daty.* Praha : Grada, 2022. 978-80-271-0659-2.

**Sakr, Sherif. 2020.** *Big Data 2.0 Processing Systems,A Systems Overview, Second edition.* New York City : Springer, 2020. 978-3-030-44186-9.

**Seidl, Petr. 2003.** ADOC.PUB. *Význam a způsoby sdílení geodat.* [Online] 2003. [Citace: 5. Leden 2024.] https://adoc.pub/vyznam-a-zpsoby-sdileni-geodat-ing-petr-seidl-csc-arcdata-pr.html#:~:text=Geodata%20%E2%80%A2%20data%20s%20implicitn%C3%ADm,jev%C5%AF%20a%20hranic%20mezi%20nimi.

**Simply Rhino. 2024.** Simply Rhino 3D Modeling Software Grasshopper. *Simply Rhino.* [Online] 2024. [Citace: 26. Únor 2024.] https://simplyrhino.co.uk/3d-modelling-software/grasshopper.

**Souček, Jan. 2021.** *I geografie má velká data.* [Webový článek] Praha : VESMÍR, spol. s r. o., 2021. 1214-4029.

**Večeřa, Martin. 2024.** *Technologické trendy v roce 2024 – Váš nezbytný průvodce.* [Webový článek] Brno : Lumeer.io s.r.o, 2024.

**WorldWind Community Edition, contributors. 2022.** Github WorldWindEarth WorldWindKotlin. *WorldWindKotlin.* [Online] 2022. [Citace: 22. Únor 2024.] https://github.com/WorldWindEarth/WorldWindKotlin.