

# Ukládání rozsáhlých dat v NoSQL databázích

František Koleček, [xkolec08@stud.fit.vut.cz](mailto:xkolec08@stud.fit.vut.cz)

Tomáš Moravčík, [xmorav41@stud.fit.vut.cz](mailto:xmorav41@stud.fit.vut.cz)

David Sladký, [xsladk07@stud.fit.vut.cz](mailto:xsladk07@stud.fit.vut.cz)

zima 2022

# Obsah

<b>I</b>	<b>Analýza zdrojových dat a návrh jejich uložení v NoSQL databázi</b>	<b>1</b>
<b>1</b>	<b>Analýza zdrojových dat</b>	<b>2</b>
<b>2</b>	<b>Návrh způsobu uložení dat</b>	<b>3</b>
2.1	File . . . . .	3
2.1.1	PRECEDES . . . . .	3
2.1.2	DEFINES . . . . .	3
2.1.3	CANCELS . . . . .	3
2.2	PA . . . . .	4
2.2.1	SERVER BY . . . . .	4
2.2.2	RELATED . . . . .	4
2.2.3	GOES IN . . . . .	4
2.2.4	IS IN . . . . .	4
2.3	TR . . . . .	5
2.4	Station . . . . .	5
2.5	Day . . . . .	6
<b>3</b>	<b>Zvolená NoSQL databáze</b>	<b>7</b>
<b>II</b>	<b>Návrh, implemetace a použití aplikace</b>	<b>8</b>
<b>4</b>	<b>Návrh aplikace</b>	<b>9</b>
4.1	Zpracování souborů XML . . . . .	9
4.1.1	Struktura XML souborů . . . . .	9
4.1.2	Zvolené Prostředky . . . . .	9
4.2	Stahovanie dát . . . . .	10
4.2.1	Způsob implementace . . . . .	10
4.3	Práce s databází . . . . .	10
<b>5</b>	<b>Způsob použití</b>	<b>11</b>
5.1	Neo4j . . . . .	11
5.1.1	Databáze na cloudu . . . . .	11

5.1.2	Lokální databáze . . . . .	11
5.2	Spuštění . . . . .	11
<b>6</b>	<b>Experimenty</b>	<b>13</b>
6.1	Inicializace databáze . . . . .	13
6.2	Dotazování . . . . .	14
6.3	Rychlost zpracování XML souborů . . . . .	15

## Část I

# Analýza zdrojových dat a návrh jejich uložení v NoSQL databázi

# Kapitola 1

## Analýza zdrojových dat

Použitá datová sada se nachází na stránkách ministerstva dopravy a to konkrétně zde. Detailní popis formátu dokumentů datové sady lze najít na stejném místě.

Datová sada se skládá z XML souborů. Tyto soubory jsou zveřejňovány na začátku roku pro celý rok ve složce GVD. Dále je pak každý měsíc zveřejňována datová sada pro daný měsíc s aktualizacemi pro spoje. Každý soubor má element **CZPTTCreation**, který určuje jeho vytvoření.

XML soubory lze rozdělit do následujících tří skupin:

- Definující spoj
- Rušící spoj
- Definující náhradní spoj

Soubory definující spoje mají jako kořenový element **CZPTTCISMessage**. První důležité informace se nachází v elementu **Identifiers**, kde jsou uvedeny identifikátory pro definované spojení a vlak, který ho bude provádět. Dále element **CZPTTHeader** určuje, zda spoj přijíždí nebo pokračuje z/do zahraniční stanice. Elementy **CZPTTLocation** obsahují jednotlivé stanice, kterými vlak projíždí. Zde jsou uvedeny i další informace ke stanici. Nejvýznamnější z nich jsou: čas příjezdu/odjezdu, typ aktivity. Po uvedení všech stanic následuje element **PlannedCalendar**, který určuje výčet dní, kdy je tento spoj prováděn.

Soubory rušící spoje mají kořenový element **CZCanceledPTTMessage**. Podobně jako soubory definující spoje obsahují identifikaci spoje, který se ruší a výčet dní, kdy se ruší. Dále už neunesou žádné informace.

Soubory definující náhradní spoje mají stejnou strukturu jako soubory definující spoje jenom s jediným rozdílem. Obsahují element **RelatedPlannedTransportIdentifiers**, který určuje, jaký spoj nahrazují. Tyto spoje mají unikátní identifikátor vůči normálním spojům.

## Kapitola 2

# Návrh způsobu uložení dat

Z datové sady poskytnuté XML soubory lze definovat několik entit a vazeb mezi nimi vhodných pro uložení do NoSQL databáze.

### 2.1 File

Samotné soubory lze považovat za entity pro uložení. Tyto entity ponesou informace o jméně souboru, který reprezentují a času, kdy byl soubor vytvořen z elementu CZPTTCreation.

:File
filename
creation

#### 2.1.1 PRECEDES

Novější soubory aktualizují stav definovaný předcházejícími soubory, proto mezi jednotlivými entitami :File budou vazby :PRECEDES, které spojí soubor s jeho nejbližším následníkem.



#### 2.1.2 DEFINES

Pokud se nejedná o soubor rušící spoj, tak tento soubor definuje o jakém spoji podává informace. Tuto vlastnost bude modelována vazbou :DEFINES.



#### 2.1.3 CANCELS

Pokud se jedná o soubor rušící spoj, tak tento soubor definuje jakého spoje ruší jízdy. Tuto vlastnost bude modelována vazbou :CANCELS.

:File – :CANCELS -> :PA

## 2.2 PA

Další významnou entitou jsou spoje. Tyto spoje budou označovány jako PA – stejně jako v souborech v elementu Object Type.

:PA
Core
Company
Variant
TimetableYear
NetworkSpecificParameters
Header

### 2.2.1 SERVER BY

Každý spoj je prováděn vlakem. Tento vlak je vždy uveden v souboru se spojem, který provádí. Tuto vlastnost definujeme jako :SERVED BY.

:PA – :SERVED BY -> :TR

### 2.2.2 RELATED

Pokud se jedná o spoj, který nahrazuje jiný spoj, tak je také definováno jaký spoj je nahrazován. Tuto vlastnost modelujeme jako :RELATED.

:PA – :RELATED -> :PA

### 2.2.3 GOES IN

U každého spoje je také uvedeno, v jaké dny jezdí. Informaci v jaké dny jeden reprezentujeme vazbou :GOES IN mezi spojem a dnem.

:PA – :GOES IN -> :Day

### 2.2.4 IS IN

Každý spoj projíždí minimálně dvěma stanicemi. Tuto vlastnost modelujeme vztahem :IS IN mezi spojem a stanicí. Tato vazba také neve řadu parametrů definující příjez, odjezd, prováděné akce a podobné.

:PA – :IS IN -> :Station

:IS IN
LocationSubsidiaryCode
AllocationCompany
LocationSubsidiaryName
ALA
ALAOffset
ALD
ALDOffset
DwellTime
ResponsibleRU
ResponsibleIM
TrainType
TrafficType
CommercialTrafficType
TrainActivityType
OperationalTrainNumber
NetworkSpecificParameters

## 2.3 TR

Soubory jsou také definovány vlaky, které provádí spoje. Tyto vlaky budou znázorněny entitami :TR.

:TR
Core
Company
Variant
TimetableYear
NetworkSpecificParameters

## 2.4 Station

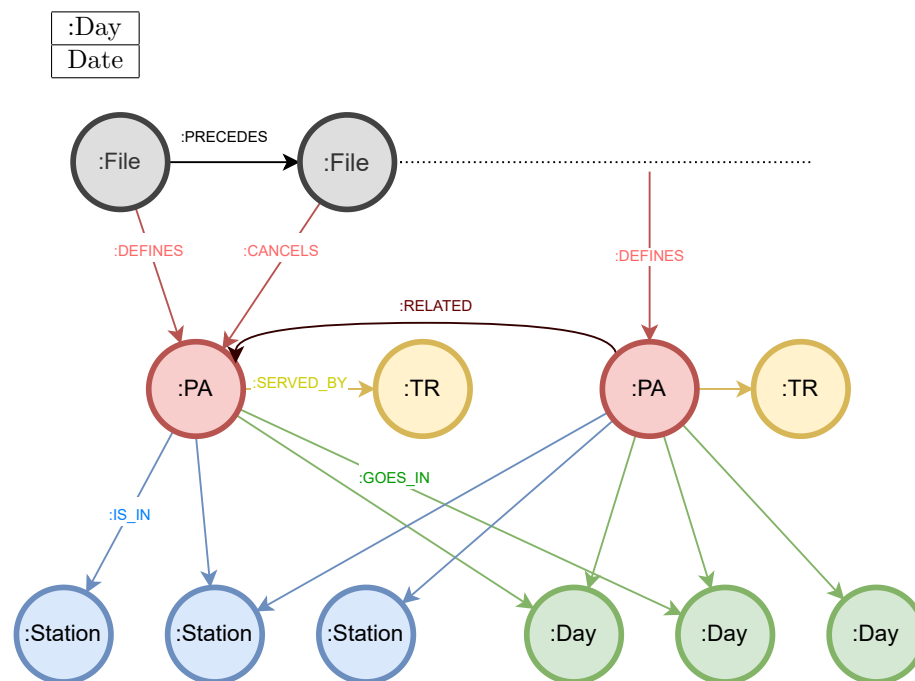
V XML souborech jsou také definovány stanice, kterými vlak projíždí. Tyto stanice budou modelovány pomocí entit :Station.

:Station
LocationPrimaryNumber
PrimaryLocationName
CountryCodeISO



## 2.5 Day

U každého spoje je také určeno, v jaké dny jezdí. Tyto dny budou existovat jako entity :Day.



Obrázek 2.1: Finální schéma entit a vazeb mezi nimi v databázi.

## Kapitola 3

# Zvolená NoSQL databáze

Pro řešení tohoto problému jsme zvolili grafovou databázi, konkrétně Neo4j. Grafová databáze nám umožní reprezentovat entity jako uzly v grafu a vztahu mezi entitami jako cesty mezi těmito uzly. Zároveň je možné u uzlů i cest definovat další vlastnosti typu klíč:hodnota, co zase umožní uložit veškeré další potřebné informace vztahující se k jednotlivým objektům. Neo4j poskytuje webové rozhraní pro databázi, které umožňuje jednoduché zadávání požadavků na databázi. S kombinací s tím, že toto webové rozhraní také zobrazuje získané výsledky do grafu, tak nám to umožní snadnější opravování chyb v aplikaci a snadnější vytváření potřebných požadavků na databázi.

## Část II

# Návrh, implemetace a použití aplikace

## Kapitola 4

# Návrh aplikace

### 4.1 Zpracování souborů XML

Informace o vlakových spojích jsou získávány ze souborů ve formátu XML. Tyto soubory je třeba zpracovat – nahrát data do strukturované vnitřní reprezentace programu, pro efektivní nahrávání do databáze. Každý soubor obsahuje informace o jednom konkrétním vlakovém spoji, jeho cesta a časy ve stanicích jsou vždy stejné, jsou zde definované dny, ve kterých tento spoj jede. Nejdůležitější zpracovávaná data jsou:

- Identifikátory
- Navštívené stanice a časy příjezdu a odjezdu
- Dny, ve kterých spoj jede

#### 4.1.1 Struktura XML souborů

Každá s těchto informací se nachází ve vlastní „větvi“ v souboru. Nachází se u nich samozřejmě i dodatečné informace – detaily lokace, činnost vlaku ve stanici atd. Platné dny jsou určeny pomocí dvou atributů – začátek a konec platnosti a bitmapa. Bitmapou je myšlen řetězec jedniček a nul, kde jedničky vyjadřují platnost v jednotlivých dnech. Tímto způsobem lze vyjádřit platné dny na rok dopředu pomocí řetězce dlouhém 365 znaků.

#### 4.1.2 Zvolené Prostředky

Zpracovávání souborů je implementováno v jazyce Python v souboru `parser.py`. Je využito modulu `xml`, který je součástí základní instalace Pythonu, není třeba jej dodatečně instalovat. Tento modul obsahuje třídu `ElementTree`, která umožňuje snadné nahrávání dat ze stromové struktury souboru. Velmi užitečnou funkcí je možnost adresování uzlů pomocí „cesty“ - obdobným způsobem jako adresování souborů v souborovém systému.

## 4.2 Sťahovanie dát

Súbory sú stiahnuté prostredníctvom súboru `download.py` v jazyku Python. Bola využitá knižnica `BeautifulSoup4`, ktorá slúži na získavanie dát z HTML súborov. Ďalším podstatným modulom je `requests`, využitý na volanie GET požiadavkov na stiahnutie zip súborov. Súbor `download.py` obsahuje dve triedy; `DataDownload` a `DataExtract`. Pomocou prvej triedy sa stiahnu všetky doposiaľ nestiahnuté súbory náležiacie danému roku jízdných radov a uložia sa do priečinku `/zips`. Druhou triedou sú tieto súbory extrahované už v podobe xml súborov, pripravených na ďalšie spracovanie.

### 4.2.1 Způsob implementace

Nejdůležitější roli při zpracovávání hraje funkce `node_to_dict`, která rekurzivně prohledává daný uzel a převádí jeho obsah na slovníkový datový typ.

Protože platné dny jsou v naší databázi ukládány jako vlastní uzly, na které pak odkazují spoje, které jsou platné v daný den, je potřeba původní reprezentaci platných dní (popsána výše) převést na seznam konkrétních kalendářních dat. Pro implementaci této funkcionality byl využit modul `datetime`, který je opět součástí základní instalace pythonu. Tento modul mimo jiné umožňuje vykonávat „aritmetické operace“ s kalendářními daty. V našem případě se jedná o sečtení data začátku platnosti a indexu jedničky v příslušné bitmapě. Implementace je ve funkci `cal_to_listofdays`.

## 4.3 Práce s databází

Obsluhování databáze je implementováno v modulu `database.py`. Obsahem tohoto module je třída `Database` s metodami obsahující query na spravování entit (File, PA, TR, Day, Station), tak vazby mezi nimi (PRECEDES, DEFINES, CANCELS, GOES IN, IS IN, RELATED). Stěžejní query je však pro vyhledávání spoje:

```
MATCH (s:Station)<-[i:IS_IN]-(p:PA)-[ii:IS_IN]->(ss:Station)
MATCH (p)-[:GOES_IN]->(d:Day)
MATCH (sx:Station)<-[ix:IS_IN]-(p)-[ii:IS_IN]->(ss:Station)
WHERE s.PrimaryLocationName="Mosty u Jablunkova"
AND ss.PrimaryLocationName="Ostrava hl.n."
AND time(i.ALA) > time("00:00:00")
AND time(ii.ALA) > time(i.ALA)
AND d.Date="2022-03-31"
AND time(ix.ALA) > time(i.ALA)
AND time(ix.ALA) < time(ii.ALA)
AND '0001' IN i.TrainActivityType AND i.TrainType='1'
AND '0001' IN ii.TrainActivityType AND ii.TrainType='1'
RETURN s.PrimaryLocationName, i.ALA, sx.PrimaryLocationName,
       ix.ALA, ss.PrimaryLocationName, MIN(ii.ALA)
```

## Kapitola 5

# Způsob použití

### 5.1 Neo4j

Pro spuštění je potřeba mít přístup k Neo4j databázi. Toho se dá dosáhnout několika způsoby, zde však budou uvedeny jenom dva nejpřímochařejší. Je také potřeba mít v Pythonu doinstalovaný modul Neo4j: `pip install neo4j`. Pro správnou funkčnost tohoto module je potřeba Python alespoň verze 3.6.

#### 5.1.1 Databáze na cloudu

Pro tuto variantu je potřeba si založit účet na stránkách Neo4j -> Get Started -> Start Free -> Sign up. Po vytvoření účtu je možné se dostat ke správci cloudových databází obdobným způsobem opět z Neo4j -> Get Started -> Start Free -> Přihlášení. Po přihlášení lze vytvořit novou databázi pomocí New Instance. Nezapomeňte si poznamenat přihlašovací údaje do databáze.

#### 5.1.2 Lokální databáze

Pro lokální databázi je třeba stáhnout Neo4j Community. Po rozbalení staženého archivu zadejte následující příkaz z kořenového adresáře: `./bin/neo4j console`. Databáze bude poskytovat webové rozhraní na `http://localhost:7474/`. Výchozí login i heslo jsou `neo4j`.

### 5.2 Spuštění

```
python3 app.py login heslo uri
```

- `login` - přihlašovací jméno k databázi
- `heslo` - heslo k databázi
- `uri` - uri databáze

Po spuštění programu se bude program dotazovat na několik informací:

1. Do you want to download data? Y/N - Tato volba způsobí stažení zipovaných xml do složky zips a jejich následovné rozbalení do složky `xml_data`. Pokud nechcete stahovat data, umístěte vlastní xml soubory do `xml_data`.
2. Do you want to load local data? Y/N - Projde soubory v `xml_data` a nahraje jejich grafovou reprezentaci do databáze.
3. Please put starting station - Zadejte výchozí stanici.
4. Please put terminal station - Zadejte konečnou stanici.
5. Please put since date - Zadejte čas od kdy má program vyhledávat ve formátu YYYY-MM-DDTHH:MM:SS. T je oddělovač data a času.

Po zadání všech informací program vyhledá nejbližší přímý spoj, pokud takový existuje, mezi stanicemi. Posléze budete dotázáni, zda chcete provést další vyhledávání.

Příklad výstupu programu.

```
user@pc:/path/to/root/folder$ python3 app.py neo4j neo4j bolt://localhost:7687
Do you want to download data? Y/N
n
Do you want to load local data? Y/N
n
Please put starting station:
Polanka nad Odrou
Please put terminal station:
Sedlnice
Please put since date:
2022-07-13T9:00:00
-----
Polanka nad Odrou -- 09:43:00
Jistebník -- 09:46:00
Studénka -- 09:51:00
Sedlnice-Bartošovice -- 09:56:00
Sedlnice -- 09:57:00
It took 0.004521846771240234 s to complete this query.
-----
Do you wish to continue?
n
```

## Kapitola 6

# Experimenty

Experimenty byly prováděny na přístroji s tímto HW/SW:

- OS: Pos os 22.04
- CPU: Ryzen 5 5600X
- RAM: 16 GB

Experimenty byly prováděny vůči lokálně běžící databázi.

### 6.1 Inicializace databáze

Počet spracovaných souborů	13 051
Doba provedení	22,9 h
Vytvořených GOES IN vazeb	5 781 132
Vytvořených IS IN vazeb	552 858



## 6.2 Dotazování

Výchozí stanice	Ropice zastávka
Cílová stanice	Dolní Lutyně
Datetime	2022-03-31T00:00:00
Doba provedení	1.1645660400390625s
Výsledek	Ropice zastávka – 17:36:00 Český Těšín nákl.n. – 17:38:00 Český Těšín – 17:39:00 2.TK loucká – 17:44:30 Chotěbuz (Kocobědz) – 17:45:00 Louky nad Olší – 17:47:30 Karviná-Darkov – 17:55:00 Stan. Karviná-Darkov – 17:55:30 Karviná hl. n. – 17:58:00 Koukolná odbočka – 18:01:00 Dětmarovice – 18:03:00 Dolní Lutyně – 18:06:30

Výchozí stanice	Mosty u Jablunkova
Cílová stanice	Ostrava hl.n.
Datetime	2022-03-31T00:00:00
Doba provedení	1.1052453517913818
Výsledek	No connection found.

Výchozí stanice	Mosty u Jablunkova
Cílová stanice	Ostrava hl.n.
Datetime	2022-03-31T00:00:00
Doba provedení	1.1052453517913818
Výsledek	No connection found.

Výchozí stanice	Rajhrad
Cílová stanice	Rajhrad
Datetime	2022-06-06T00:00:00
Doba provedení	0.6875042915344238s
Výsledek	No connection found.

Výchozí stanice	Rajhrad
Cílová stanice	Brno-Královo Pole
Datetime	2022-10-24T00:00:00
Doba provedení	0.01873946189880371s
Výsledek	No connection found.

Výchozí stanice	Rajhrad
Cílová stanice	Brno hl.n.
Datetime	2022-24-10T00:00:00
Doba provedení	0.23762035369873047s
Výsledek	No connection found.

### 6.3 Rychlost zpracování XML souborů

Funkčnost a efektivita skriptu `parser.py` byla testována nad složkou `GVD2022`, která obsahuje přibližně 12300 souborů formátu XML. Všechny soubory byly v experimentu zpracovány a výpis důležitých informací vytisknut na standardní výstup. Tato operace trvala 93 sekund, tedy cca 132 zpracovaných souborů za sekundu.