Projektová dokumentácia UPA

Ondrej Valo, xvaloo00@stud.fit.vutbr.cz
Radoslav Páleník, xpalen05@stud.fit.vutbr.cz
Karel Fritz, xfritz00@stud.fit.vutbr.cz

zima 2022

Obsah

Část I

Analýza zdrojových dat a návrh jejich uložení v NoSQL databázi

Kapitola 1

Analýza zdrojových dat

Pro potřeby projektu byla poskytnuta archivovaná data z portálu *Celostátního Informačního Systému o jízdních řádech* (CIS JR) uchovávající informace o celostátních vlakových spojeních v České republice platných pro kalendářní rok 2022.

1.1 Poskytnuté data

Nosným zdrojem je archiv *GVD2022.zip*, obsahující informace o každém vytvořeném jízdním řádu pro spoj ve formě *.XML* souboru. Jízdním řádem rozumíme jednotlivé zastávky daného vlakového spoje jedním směrem. Uvedené jízdní řády lze reprezentovat v modelu DOM následovně:

CZPTTCISMessage - objekt udržující informace o daném spoji.

- *Identifiers* - Seskupení identifikátorů daného spojení
- *CZPTTInformation* - Informace o daném spoji
  - *CZPTTLocation* - Zastávka spoje v dané lokalitě
    - *Location* - Informace o lokalite zastávky
    - *TimingAtLocation* - Určení času pro spoj v dané lokalitě
- *NetworkSpecificParameter* - Identifikátory jednotlivých přepravců

1.2 Aktualizace pro poskytnuté data

Kromě poskytnutých dat v *GVD2022.zip* je na portálu uveden archiv s opravou poznámek pro jednotlivé, již vytvořené vlakové spoje. Jedná se o data uvedená ve stejném formátu, jako původní soubory. Zdrojový adresář se zmíněnými archivy obsahuje také skupinu záznamů s prefixem *cancel_*, podle které se uvádějí rušení jízdního řádu v uvedené kalendářní dny na základě identifikace spoje *PA ID* z *CZPTTCISMessage.Identifiers*.

Kapitola 2

Návrh způsobu uložení dat

Data jsou původem z portálu "https://portal.cisjr.cz/pub/draha/celostatni/szdc/2022/", obsahují spoje Českých vlakových spojů. Data jsou strukturovaná a usnadňují tím způsob uložení a yacházení s nimi.

2.1 Získání dat

K získání dat byla použita knihovna requests, konkrétně její metoda get(). Metoda get() přijímá jako parametr konkrétní url webové stránky jejíž obsah má stáhnout. Stahování dat je provedeno paralelně, pro vyšší rychlost. Konkrétně třída ProcessingThread.

2.2 Zpracování dat

Při získávání dat bylo třeba zavést oddělené kolekce. Jednotlivé názvy kolekcí jsou CZCanceledPTTMessages, CZPTTCISMessages, CZUpdatedPTTMessages, přičemž hlavní je CZPTTCISMessages. Kolekce CZPTTCISMessages představuje originálně naplánované spoje vlaků. Kolekce CZCanceledPTTMessages obsahuje spoje zrušené a kolekce CZUpdatedPTTMessages spoje upravené. Jiné zpracování nebylo potřebné vzhledem k strukturovanosti dat, se kterými pracujeme.

2.3 Uložení dat do databáze

Manipulace s položkami databáze je implementována ve třídě Dataset. Metoda přidávající data do databáze je metoda insert. Také tato metoda pracuje paralelně. Jako vstup přijímá názvy souborů s daty pro databázi. Data jsou do databáze nahrána bez úprav schématu, struktura položek zůstala zachována.

Kapitola 3

Zvolená NoSQL databáze

3.1 Volba databáze

Mezi možné řešení jsme původně uvažovali různé databáze například Apache Cassandra, firebase, redis, mongoDB. V závěru vyhrála databáze mongoBD a to hlavně díky její jednoduchosti a dobré dokumentaci. MongoDB uplatňuje "Consistency"a "Partition tolerance"v CAP teorému, vynecháním "Availability". Pracuje dobře s programovacím jazykem python a tedy tato dvojice je tedy základ pro náš projekt. Python programovací jazyk, pracující nad mongoDB.

Část II

Návrh, implemetace a použití aplikace

Kapitola 4

Návrh aplikace

4.1 Struktura aplikace

Aplikace je rozdělena do dvou úrovní. Základní soubor obsahující kód s přímým napojením na databázi. Ostatní dva soubory jsou pro kontrolování chodu programu, přijímají argumenty a podle nich přizpůsobí i celý běh aplikace.

4.2 dataset.py

Soubor dataset.py obsahuje třídu Dataset, ta je přímo napojena na naši mongoDB hostované na AWS. Třída Dataset obsahuje metody init, clear, insert a chunks. Init je spuštěna první a zde se také provede napojení na vzdálenou databázi.

4.3 download.py

Dalším souborem je download.py, řídící akce a obsahující více možností pro vstup. Je zde tedy "argument parser", u kterého volíme –url, –download, –updata, –cancel_update, –clear. Možnost clear vymaže všechny kolekce. Při volbě příslušného url a možnosti –download následuje stažení záznamů, které musí být také rozbaleny (na to slouží funkce getlist_of_names_from_gzips() a get_list_of_names_from_gzips().

4.4 lookup.py

Poslední soubor lookup.py obstarává funkci vyhledání cesty z místa A na místo B. Pro správnou funkci je důležité mít vyplněny patřičné argumenty programu, tedy –time, –from_city, –to_city. Za –from_city a –to_city můžeme považovat lokaci, ve které vlak dělá zastávku. Navíc nalezená cesta musí obsahovat spoje jedoucí pouze později než je zadaný.

Kapitola 5

Způsob použití

Používání aplikace je majoritně k vyhledávání spojů. Pro to je speciální typ dotazu, před touto funkcionalitou je však třeba nejdříve data nahrát.

5.1 Stažení dat

Data se do vzdálené databáze dostanou spuštěním programu download.py, konkrétně s argumenty –url a –download. Argument –url je defaultně nastavený na adresu "https://portal.cisjr.cz/pub/draha/celostatni/szdc/2022/", tudíž není třeba ho specifikovat. Druhý argument –download pouze nastavuje True hodnotu pro args.download. Stažení dat a nahrátí do DB provede následující příkaz: python3 download.py –url –download

5.2 Update dat

Při updatu dat je třeba program spustit s argumentem –update, program nejříve stáhne nové spoje z url, které je opět nastaveno na defaultní hodnotu, potom provede update hodnot v databázi. Samotné rozdělení funkcí insert a update je až v samotné metodě insert třídy dataset. Příkaz, jež provede update DB: python3 download.py –url –update

5.3 Vyhledání trasy

Vyhledání trasy je hlavní funkcí implementované aplikace. Pro její spuštění je důležité programu na vstupu zadat argumenty –time, –from_city, –to_city. Tyto tři argumenty tvoří omezení pro výstup dotazu, ten totiž tvoří trasu s jednotlivými přestupy. Jednotlivé přestupy, zastávky a odjezdy jsou všechny v čase budoucím od zadaného argumentu –time. Vzorový příkaz pro vyhledání trasy z Zittau do Chotyně: python3 lookup.py -t 2021/09/17-16:20:00 -from Zittau -to Chotyně

Kapitola 6

Experimenty

Databáze běží na výpočetních serverech Amazon Web Services. Použitá NOSql databáze je MongoDB.

6.1 Nahrání dat / Update dat

Nahrávání dat bylo v prvních verzích programu, mnohem pomalejší a trvalo 3h. Tento problém se podařilo z velké části vzřešit paralelním během. Po paralelizaci bylo dosáhnuto zrychlení, potřebný čas byl pouze 20%. Tedy konečný čas nahrávání při použití paralelizace je 36min.

Update záznamů ze souboru 'GVD2022-oprava_poznamek_KJR_vybranych_tras20220126.zip' promítnut do DB trvá 1min, opět při použití paralelizace.

Cancel_update stáhne kolekce a uploadne je do databáze, tato možnost, opět s paralelizací trvá asi 4min. Důvodem pomalejšího běhu je pravděpodobně zabalení do formátu gzip samostatně, každý soubor je třeba nejdřív rozbalit a pak až číst.

6.2 Dotazy

Dotaz na vyhledání cesty trvá průměrně 1-3sec, záleží na počtu nalezených spojů a délce trasy.