UPA - Projekt, 1. část: ukládání rozsáhlých dat v NoSQL databázích

Timotej Ponek, xponek00

25. října 2022

Obsah

Časť I

Analýza zdrojových dát a návrh ich uloženia v NoSQL databázi

Kapitola 1

Analýza zdrojových dát

Zdrojové súbory sú distribuované vo formáte, ktorý je zdokumentovaný v súbore správy železniční dopravní cesty [1]. Ide o súbory vo formáte .xml. V zásade existujú dva typy súborov:

- *CZPTTCISMessage* - Ide o súbor, ktorý vytvára nové jízdní řády, pridávané jízdní řády sú uložené v elemente *CZPTTInformation*. Ďalej je pre nás zaujímavý element *Identifiers*, skrz ktorý je správa o vytvorení jízdní řádu (*CZPTTCISMessage*) prepojená so správou o zrušení jízdní řádu (*CZCanceledPTTMessage*). Ostatné elementy (*CZPTTCreation, CZPTTHeader, NetworkSpecificParameter*) sú pre funkcionalitu požadovanú v zadaní nepodstatné.
- *CZCanceledPTTMessage* - Ide o súbor, ktorý vytvára výluku v jízdom řáde, na určitú dobu, ktorá je zistiteľná z elementu *PlannedCalendar*. Výluka môže byť iba na určitej časti trate, alebo na trati celej.

Štruktúra elementu *CZPTTInformation*: Element *CZPTTInformation* je pole, obsahujúce objekty identifikujúce lokácie, v ktorých sa vlak bude počas vykonávania jízdního řádu nachádzať, v aký čas sa v nich má nachádzať a akú aktivitu bude práve vykonávať. Vymenované informácie sa nachádzajú v elemente *CZPTTLocation*. Tento element obsahuje viacero podelementov, z ktorých sú pre nás (pre použitie aplikácie vyžadované zadaním) zaujímavé iba nižšie uvedené:

- *Location* - Element identifikujúci lokáciu, obsahuje názov lokácie.
- *TimingAtLocation* - Obsahuje čas, v ktorý má vlak doraziť na danú lokáciu.
- *TrainActivity* - Pole ktoré identifikuje aktivity, ktoré vlak vykonáva v danej lokácií (napríklad že zastavuje v danej lokácií, alebo zastavuje iba na znamenie...)

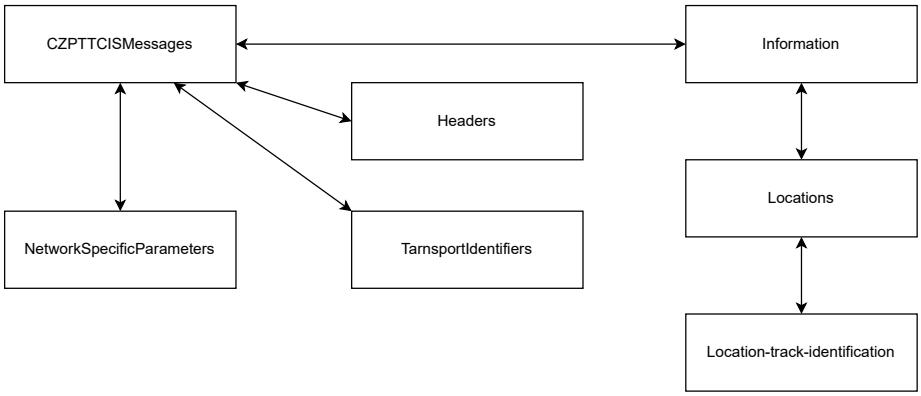
[1] `https://portal.cisjr.cz/pub/draha/celostatni/szdc/Popis%20DJ%C5%98_CIS_v1_09.pdf`

Kapitola 2

Návrh spôsobu uloženia dát

Spracovanie vstupných .xml súborov: Pre implementáciu bol zvolený jazyk Java. Bola použitá knižnica *JABX*, ktorá umožňuje namapovať vstupný XML súbor na triedy definované v Jave, a teda vytvoriť zo vstupného .xml súboru objekt v Jave. Pre správne fungovanie tohto mapovania je najskôr potrebné vygenerovať schému .xml súboru z tzv. "root"objektu volaním `JAXBContext.newInstance-(CZPTTCISMessage.class)`. Toto volanie vytvorí z objektu *CZPTTCISMessage* schému pre namapovanie .xml súboru, ktorý obsahuje *CZPTTCISMessage*, a zároveň validuje, že objekt je mapovateľný do formátu .xml. Obdobne funguje vytvorenie schémy pre *CZCanceledPTTMessage*. Vstupné .xml súbory sa teda spracúvajú tak, že sú namapované na objekty v Jave pomocou volania metódy *Unmarshall()*.

Mapovanie objektov na databázové kolekcie: Pre mapovanie objektov bol zvolený framework *Morphia*. Pomocou neho sa namapovali objekty, reprezentujúce .xml správy na kolekcie, ktoré sa následne uložia do databázy. Vzniknuté kolekcie je možno vidieť na obrázku2.1 (šípky na obrázku naznačujú že jednotlivé kolekcie sú vzájomne prepojené referenciami).



Obrázek 2.1: CZPTTCISMessage

Kapitola 3

Zvolená NoSQL databáza

Bola zvolená dokumentová NoSQL databáza MongoDB. Bola zvolená hlavne z dôvodu jej jednoduchosti a popularity.

Výhody: MongoDB ukladá dáta vo forme BSON objektov, čo je formát totožný s formátom JSON. S JSON formátom som už mal predošlé skúsenosti, takže získavanie dát z databázi a následná interpretácia týchto dát bola pre mňa jednoduchá.

Časť II

Návrh, implementácia a použitie aplikácie

Kapitola 4

Návrh aplikácie

Rozdelenie do balíčkov: Aplikácia je členená do balíčkov. Hlavný balíček je `UPA.main`, ktorý obsahuje funkciu *main* - štartovací bod aplikácie. Ďalšie balíčky sú:

- `UPA.XML_Schema` - Obsahuje triedy *CZPTTCISMessage* a *CZCanceledPTTMessage*, ktoré reprezentujú vstupné .xml súbory, a ďalšie triedy ktoré reprezentujú jednotlivé elementy vstupných .xml súborov.
- `UPA.XML_Loader` - Obsahuje triedu *Downloader*, ktorá ponúka metódy na stiahnutie .xml súborov z webu `https://portal.cisjr.cz/pub/draha/celostatni/szdc/`, a spracovanie týchto súborov - konverziu z .xml do Java objektov a následné uloženie týchto objektov pomocou Morphia frameworku do databázy.
- `UPA.MongoDB_Client` - Obsahuje triedu *MongoDB_ Client*, ktorej funkcionalita je nájsť spojenie v zadanom dni a čase, zo zadanej východzej stanice do zadanej cieľovej stanice.

Implementácia sťahovania súborov: Na získavanie jednotlivých názvov súborov bola použitá knižnica *Jsoup*, ktorá obsahuje funkcie na získanie html elementov s href atribútmi. Názvy súborov (Href atribúty) ktoré sa končia na ".zip"sa následne sťahujú a priamo ukladajú do adresára zips.

Implementácia agregačnej funkcie: Na implementáciu agregačnej funkcie bol použitý nástroj MongoDB Compass, ktorý dokáže zkonvertovať query v MongoDB syntaxi do iných programovacích jazykov (napr. Java, Python, Go). Agregačná pipline prebieha nad kolekciou *"Information"* a skladá sa z násladujúcich podfunkcií:

- $match - Zisťuje či hodnota *plannedCalendar.validityPeriod.StartDateTime* je menšia ako zadaný dátum a čas a hodnota *"plannedCalendar.validityPeriod.EndDateTime"* je vačšia ako zadaný dátum a čas, teda zjednodušene,

či zadaný dátum a čas leží v intervale <StartDateTime, EndDateTime>. Ďalej zisťuje či *czpttLocations.trainActivities.TrainActivityType* je zhodný s "0001", teda či vlak zastavuje v danej stanici.

- $set - Nastavuje nové políčko *IsDayValid* vo výslednej kolekcii, ktoré reprezentuje či je jízdní řád platný v zadané datum a čas.
- $lookup - Získava z kolekcie *Locations* lokácie, ktoré sú referncované idčkom z *czpttLocations.location.$id*, a získané lokácie ukladá do nového poľa *location*.
- $match - Tento $match filtruje spracovávané objekty podľa kritérií: *location.PrimaryLocationName* je zhodné s cieľovou alebo východziou stanicou a *IsDayValid* je nastavené na "1".
- $set - Tento $set spája objekty v poli *$czpttLocations"* a v poli vytvorenom predošlým $lookup-om *locations* na základe zhody idčiek do nového poľa *locationsArray*. Spája teda názvy lokácií s ostatnými informáciami o akciách a čase výskytu vlaku v danej lokácií. Toto spájanie som sa snažil urobiť už pri prvom $lookup-e, ale nenašiel som žiadne fungujúci spôsob. Avšak spájanie dvoch polí na základe nejakej spoločnej zhodujúcej sa hodnoty (v tomto prípade idčka) funguje.
- $project - Ponecháva vo výsledku iba potrebné informácie, čo je názov lokácie a čas, v ktorom sa vlak nachádza v lokácií.

Agregačná funkcia je implementovaná skrz klasický MongoDB framework a nepoužíva Morphia framework, pretože použitie Morphia frameworku má väčšiu pamäťovú náročnosť. Morphia sa používa iba pre ukladanie namapovaných .xml objektov a v nich uložených podobjektov do databázy, pre svoju jednoduchosť v ukladní už existujúcih objektov (stačí objekt namapovať na entitu a ukladať už celý objekt, nie po jednotlivých hodnotách).

Kapitola 5

Spôsob použitia

Pre správne fungovanie aplikácie je potrebné nainštalovať MongoDB, podľa postupu ktorý je možné nájsť v dokumentácií MongoDB. Aplikácia je implementovaná v Jave, verzi 11, a je ju možné spúšťať cez IDE Intelij IDEA, alebo skrz maven. Pre spustenie skrz maven zadajte v príkazovom riadku:
mvn clean compile exec:java

Aplikácia sa následné spustí a je zadaním číselnej hodnoty je možné vybrať jednu z následjúcich akcií:

- 1 - Stiahne zazipované xml súbory do zložky zips (iba ak už nie sú stiahnuté) následne ich spracuje a transformuje do objektov v jave a tie nahrá do databázy
- 2 - Vykonáva vyhľadávanie spoja, vypýta si od užívateľa počiatočnú stanicu, cieľovú stanicu a dátum a čas, kedy sa má spoj vyhľadať, a vypíše všetky spoje ktoré vyhovujú daným kritériám
- hocijaké iné číslo - Ukončí aplikáciu.

Obmedzenia aplikácie: V aplikácií nie je implementovaná funkcionalita (vhodná agregácia), ktorá by rátala pri vyhľadávaní spojov s výlukami vytvorenými pomocou CZCanceledPTTMessage. Nie je implementovaná z dôvodu časovej tiesne, čím chcem zdôrazniť že vytvoriť danú funkcionalitu v mojom riešení nie je nemožné.

Aplikácia ďalej pri zadávaní staníc ráta so zadaním presného názvu stanice, tak ako sa vyskytuje v databázi. Preto prosím ak vám vyhľadávanie zlyhá, skontrolujte či ste názov stanice začali s veľkým písmenom a dodržali diakritiku. Aplikácia neposkytuje funkcionalitu, ktorá by vypísala na výstup názvy všetkých staníc z databázy (znovu z dôvodu časovej tiesne a s očakávaním, že sa bude testovať s názvami staníc zhodujúcimi sa s názvami staníc vo vstupných .xml súboroch), ale táto funkcionalita by bola určite vhodná

Experimenty: Ukladanie dát môže trvať niečo okolo 30 minút.