SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

SPARK INTEGRÁCIA S ĎALŠÍMI TECHNOLÓGIAMI (KAFKA, CASSANDRA, MONGODB)

SEMINÁRNA PRÁCA

2022 Jakub Lengvarský, Jakub Kupkovič, Peter Abel, Daniel Jančí

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

SPARK INTEGRÁCIA S ĎALŠÍMI TECHNOLÓGIAMI (KAFKA, CASSANDRA, MONGODB)

SEMINÁRNA PRÁCA

Študijný program:	Aplikovaná informatika
Predmet:	I–ASOS – Architektúra softvérových systémov
Prednášajúci:	RNDr. Igor Kossaczký, CSc.
Cvičiaci:	Ing. Stanislav Marochok

Bratislava 2022 Jakub Lengvarský, Jakub Kupkovič, Peter Abel, Daniel Jančí

Obsah

Zoznam obrázkov a tabuliek

Úvod

Pri tvorbe webových aplikácií sa využívajú rôzne technológie ktoré musia byť navzájom prepojené. Táto práca sa zaoberá práve prepojením technológií Apache Spark s Apache Kafkou, Apache Cassandrou a taktiež MongoDB. Každá technológia má svoje využitie a je na používateľovi, aby vedel ktoré a ako implementovať. Dúfame, že vďaka tejto práci bude rozhodovanie o tom kedy a čo implementovať jednoduchšie.

1 Apache Spark

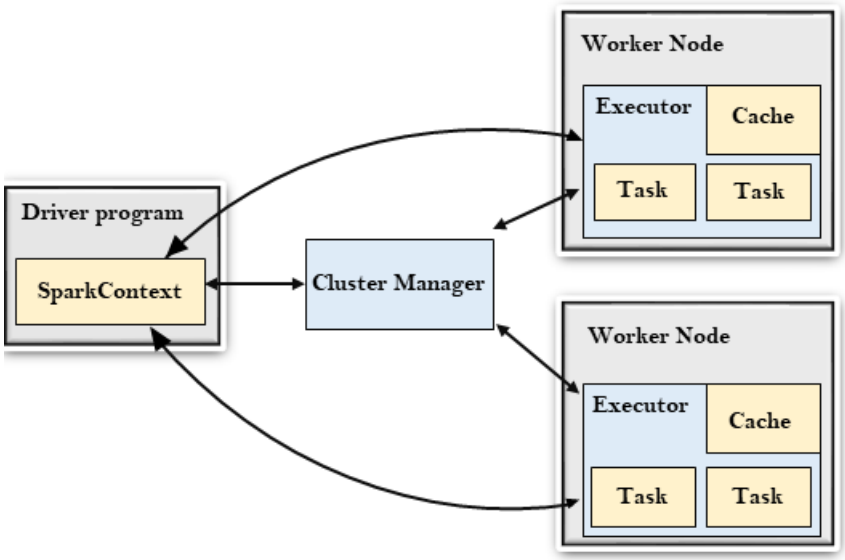
Apache Spark je open-source platforma pre distribuované výpočty, ktorá sa používa na rozsiahle spracovanie a analýzu údajov. Je navrhnutá tak, aby bola rýchla a škálovateľná, a ponúka bohatý súbor rozhraní API vo viacerých programovacích jazykoch vrátane jazykov Java, Python a Scala. Spark umožňuje vývojárom spracovávať a analyzovať veľké súbory údajov v reálnom čase.

Spark má flexibilný engine na spracovanie dát v pamäti, ktorý umožňuje spracovanie dát s nízkou latenciou a vysokou priepustnosťou, vďaka čomu je vhodný na širokú škálu pracovných záťaží vrátane iteračných algoritmov, interaktívnych dotazov a prúdových dát. Okrem toho Spark ponúka bohatý ekosystém knižníc a nástrojov, ktoré podporujú širokú škálu prípadov použitia vrátane strojového učenia, spracovania grafov a prúdového spracovania v reálnom čase.

1.1 Komponenty Apache Spark

Medzi hlavné komponenty Apache Spark patria:

- **Spark Core**: poskytuje základný exekučný engine a API pre distribuované spracovanie a analýzu dát.
- **Spark SQL**: poskytuje podporu pre štruktúrované spracovanie údajov a dotazy podobné SQL.
- **Spark Streaming**: poskytuje podporu pre spracovanie tokov v reálnom čase.
- **MLlib**: poskytuje knižnicu algoritmov strojového učenia a nástrojov na vytváranie a nasadzovanie modelov strojového učenia.
- **GraphX**: poskytuje podporu pre spracovanie a analýzu grafov.



Obr. 1: Architektúra Apache Spark

2 Apache Kafka

Apache Kafka je open-source streamovacia platforma, ktorá sa používa na vytváranie streamovacích aplikácií v reálnom čase. Je navrhnutá tak, aby bola horizontálne škálovateľná, odolná voči chybám a vysoko výkonná, vďaka čomu je vhodná pre rozsiahle dátové toky a architektúry riadené udalosťami.

Kafka je založená na modeli publikovania a odberu správ, kde producenti môžu publikovať správy do jednej alebo viacerých tém a konzumeri sa môžu prihlásiť do jednej alebo viacerých tém na príjem správ. Kafka má distribuovanú architektúru s viacerými sprostredkovateľmi, ktorí môžu byť nasadení v klastroch, čo umožňuje vysokú dostupnosť a horizontálnu škálovateľnosť.

Kafka ponúka niekoľko kľúčových funkcií, ktoré z nej robia výkonný nástroj na vytváranie tokov dát a aplikácií. Má flexibilné a rozšíriteľné rozhranie API, ktoré podporuje širokú škálu prípadov použitia vrátane spracovania údajov v reálnom čase, analýzy v reálnom čase a zasielania správ v reálnom čase. Má tiež zabudovanú podporu odolnosti voči chybám so schopnosťou automaticky replikovať údaje a obnovovať sa po zlyhaniach.

2.1 Komponenty Apache Kafka

Hlavné komponenty Apache Kafka sú:

- **Témy**: Témy sú základnou jednotkou ukladania dát v Kafke. Producenti môžu publikovať správy do jednej alebo viacerých tém a konzumenti sa môžu prihlásiť do jednej alebo viacerých tém na príjem správ. Témy sú zvyčajne organizované podľa kategórie alebo typu a môžu byť nakonfigurované tak, aby podporovali rôzne politiky uchovávania a replikácie.

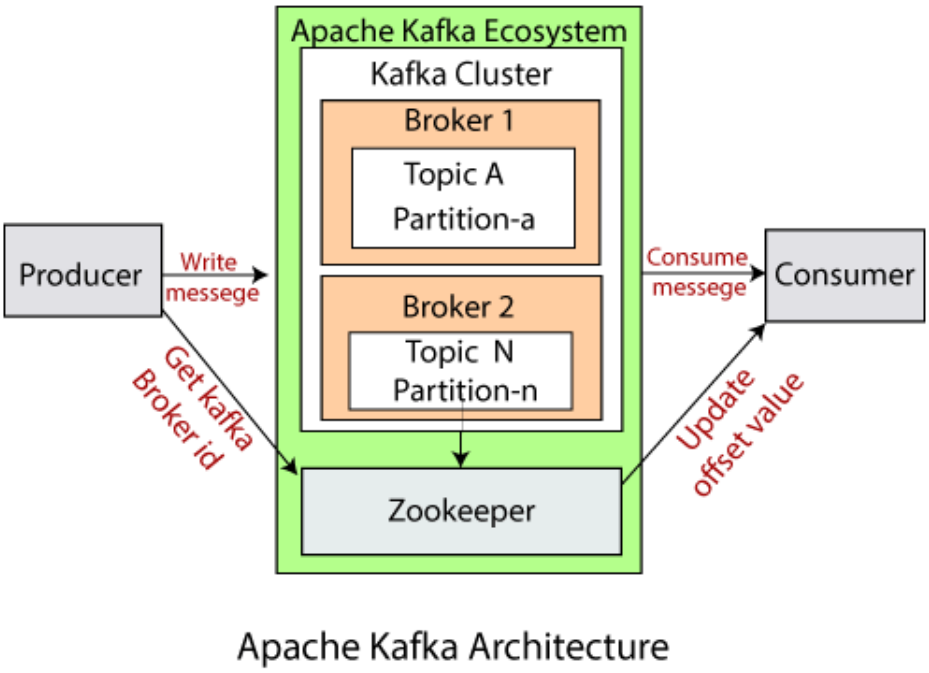
- **Brokeri**: Brokeri sú základnými komponentmi klastra Kafka a sú zodpovední za správu ukladania a distribúcie správ v rámci klastra Kafka. Brokeri udržiavajú protokol všetkých správ, ktoré sú publikované v klastri Kafka, a používajú tento protokol na doručovanie správ konzumentom a replikáciu údajov v rámci klastra.

- **Producenti**: Producenti sú klienti alebo aplikácie, ktoré publikujú správy do klastra Kafka. Producenti môžu publikovať správy do jednej alebo viacerých tém a môžu používať rôzne API a vzory správ na riadenie spôsobu doručovania správ do Kafky.

- **Konzumenti**: Konzumenti sú klienti alebo aplikácie, ktoré sa prihlásia na odber správ do jednej alebo viacerých tém v klastri Kafka. Konzumenti môžu používať rôzne rozhrania API a vzory správ na riadenie spôsobu doručovania správ z Kafky

a môžu používať údaje v týchto správach na širokú škálu účelov, napríklad na spracovanie údajov v reálnom čase, analýzu v reálnom čase a zasielanie správ v reálnom čase.

- **Zookeeper**: Zookeeper je top-level softvér vyvinutý od Apache, ktorý funguje ako centralizovaná služba a používa sa na udržiavanie konfiguračných údajov a na zabezpečenie flexibilnej a robustnej synchronizácie v rámci distribuovaných systémov. Zookeeper sleduje stav brokerov klastra Kafka a tiež sleduje témy, partície atď. V prípade zlyhania brokera dokáže Zookeeper vykonať okamžitú migráciu, napr. ak zlyhá vedúci broker, nový broker sa vyberie v reálnom čase prostredníctvom dopytovania v rámci súboru.



Obr. 2: Architektúra Apache Kafka

3 Apache Cassandra

Cassandra bola navrhnutá ako nástroj pre implementáciu rozsiahlych škálovateľnych aplikácií. Je implementovaná ako open-source program. Snaží sa byť orientovaná na dobrú dostupnosť a toleranciu voči katastrofám. Na používanie Cassandry sa využíva „Cassandra Query Language“ ktorý je špeciálne napísaný aby spĺňal požiadavky Cassandra databázy. Systémy ako Cassandra sú dizajnované na riešenie problémov ako sú napríklad:

- Úplná replikácia databázy na viacero úrovniach:
- Globalná dostupnosť a nízká latencia
- Škálovanie na klasických strojoch
- Lineárné škálovanie prietoku s každým CPU
- Rovnomerné rozkladanie výkonu a rast s počtom clusterov
- Delené kľúčovo orientované Query
- Flexibilná schema

Cassandra Query Language (CQL) organizuje dataset podľa:

- Priestoru kľúčov – Definuje ako sú datasety replikované v datacentrách. Priestory kľúčov obsahujú tabuľky
- Tabuľky – Definuje typové schémy pre kolekcie partícií. Cassandra vie dodávať flexibilne nové stĺpce bez zastavenia prevádzky.
- Partície – Definuje základnú časť primárneho kľúča. Všetky riadky v Cassandre musia mať identifikačný uzol v 1 clustery, ktorý identifikuje riadok.
- Riadok – Obsahuje súbor stĺpcov
- Stĺpec – Typ ktorý patrí riadkom.

CQL podporuje:

- Jedno partičné transakcie pomocou atomických porovnaní a setových sémantík
- Používateľom definované typy, funkcie a agregáty
- Kolekcie – sety, mapy a listy

- Sekundárne indexy

Cassandra explicitne nechce implementovať:

- Viac partičné transakcie
- Distribuované zjednotenie
- Cudzie klúče alebo referencie

4 MongoDB

MongoDB je open-source systém na správu databáz orientovaný na dokumenty, ktorý je navrhnutý tak, aby bol škálovateľný, flexibilný a ľahko použiteľný. Často sa používa na ukladanie a správu veľkého množstva neštruktúrovaných údajov, napríklad v aplikáciách, ktoré vyžadujú schopnosť ukladať a vyhľadávať zložité dátové štruktúry alebo podporovať ad-hoc dotazy a analýzu údajov v reálnom čase.

MongoDB má dátový model orientovaný na dokumenty, čo znamená, že ukladá údaje vo forme dokumentov, a nie ako riadky v tabuľke. To umožňuje ukladanie zložitých dátových štruktúr, ako sú vnorené objekty a polia, a uľahčuje vyhľadávanie a aktualizáciu jednotlivých polí v rámci dokumentu. MongoDB má tiež bohatú sadu operátorov pre dotazy a agregácie, čo uľahčuje vykonávanie komplexných analýz a transformácií údajov. Existuje niekoľko edícií:

- MongoDB Community Server — Verzia zadarmo a dostupná na Windows, Linux a MacOS
- MongoDB Enterprise Sever — Komerčná edícia
- MongoDB Atlas — On-demand Cloud verzia

Hlavné prvky:

- AD-hoc queries — Podporuje polia, query rozsahov a RegEx výrazy.
- Indexovanie — Mozu byt indexovane primárnymi aj sekondárnymi indexami
- Replikácia — Všetko sa vykonáva na primárnej kópií, ďalšie obsahujú iba kópie
- Balancovanie výkonu — Výkon sa rovnomerne rozdeľuje na všetky servery
- Súborové uložisko — Dáta sa ukladajú na viacero systémov
- Aggregácia — Využíva aggregačnú pipeline, map-redukčné funkcie, single-purpose aggregačné metódy
- Beh server-side JavaScript — Dá sa využit v Query
- Obmedzené kolekcie — Dajú sa nastaviť kolekcie s fixným počtom prvkov
- Transakcie — MongoDB tvrdí že podporuje viac dokumentové ACID transakcie

MongoDB má ale nevýhody v bezpečnosti. Počas minulých rokov sa vyskytlo niekoľko únikov informácií z MongoDB.

5 Spark Structured Streaming (Prvé demo)

Jedná sa o ukážkovú simuláciu zberu dát o počasí z meteo staníc pomocou technológií Apache Spark, Apache Kafka a Apache Cassandra. Využité boli aj technológie Python a Scala.

Aplikácia sa skladá zo 4 častí. Prvá časť je Python skript, ktorý simuluje zber dát o počasí z meteorologických staníc a následne pomocou Spark knižníc pracuje ako producent pre Kafka. Druhou časťou je Apache Kafka server, na ktorom je vytvorená téma weather, teda počasie. Tiež na ňom beží konzument. Spark aplikácia napísaná v jazyku Scala, ktorá je treťou časťou programu, využíva Kafka server ako zdroj dát. Číta ich tok, agreguje ich podľa miesta zberu meteodát a potom ich ukladá do Cassandra databázy, ktorá je poslednou časťou aplikácie. Dáta sú do databázy ukladané po batchoch, a to už po ich agregácií v Spark aplikácii.

5.1 Spustenie projektu

Je potrebné si nainštalovať Kafka, Zookeeper, Hadoop, Spark, Cassandra, Scala (2.12), Python 2 alebo 3.

Command line príkazy nižšie budú určené na konfiguráciu pre OS Windows. Najprv je potrebné pomocou CMD spustiť Zookeeper a Kafka server a nechať tieto služby spustené. Spustenie Zookeeper servera.

```
.\bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties
```

Spustenie Kafka servera.

```
.\bin\windows\kafka-server-start.bat .\config\server.properties
```

Vytvorenie Kafka topicu.

```
.\kafka-topics.bat --create --topic weather --bootstrap-server localhost:9092
```

Spustenie Cassandra servera.

```
.\kafka-topics.bat --create --topic weather --bootstrap-server localhost:9092
```

Skompilovanie Scala aplikácie a spustenie Spark aplikácie.

```
sbt package && spark-submit --class StreamHandler --master local[*]
    target/scala-2.12/stream-handler_2.12-1.0.jar
```

6 Spark integrácia s MongoDB (Druhé demo)

Druhé demo je jednoduchá implementácia connectora medzi Sparkom a MongoDB. Využité technológie boli Python (PySpark) a MongoDB.

MongoDB beží na lokálnom serveri, ktorý bolo nutné si nainštalovať a nakonfigurovať. Následne sme doňho vložili dáta tweetov, ktoré boli získané cez twitter API. Po spustení kódu sa zobrazí tabuľka s tweetmi.

6.1 Spustenie projektu

Je potrebné nainštalovať si Python, PySpark a MongoDB server. Spustenie servera:

```
sudo service mongod start
```

Spustenie python kódu:

```
python3 ./asosmongo.py
```