BigData – Projekat 2

Student: Uroš Pešić
Profesor: Dragan Stojanović

- Pomoćna aplikacija koja čita vrstu po vrstu iz emission_data.csv fajla, na osnovu kojeg se kreira DTO objekat sa odgovarajućim atributima, koji se serijalizuje u JSON i šalje na kafka topic "traffic-data".

```java
public static void produceEvent(KafkaProducer<String, DataEntryDTO> producer, String topic, String lineCsv){
        DataEntryDTO dataObj = toDTOObject(lineCsv);
        ProducerRecord<String, DataEntryDTO> rec = new ProducerRecord<>(topic, dataObj.getKey(), dataObj);
        producer.send(rec);
        System.out.println("New event. Event key(Vehicle ID) = " + dataObj.getKey());
}
public static void main(String[] args) {
        if (args.length < 2) {
        System.out.println("Missing input parameters...");
        System.exit(1);
        }
        String inputPath = args[0];
        String topic = args[1];
        try (KafkaProducer<String, DataEntryDTO> producer = createProducer()) {
                File f = new File(inputPath + "/emission_data.csv");
                try (FileReader fr = new FileReader(f)) {
                try(BufferedReader rd = new BufferedReader(fr)){
                        //skipping 1st line
                        rd.readLine();
                        String line = rd.readLine();
                        while(line != null){
                                produceEvent(producer, topic, line);
                                line = rd.readLine();
                                Thread.sleep(1000);
                        }
                producer.close();
                }
        }
    }
}
```

Atributi klase DataEntryDTO

- Iako se ne koriste svi navedeni atributi ostavio sam neke po izboru.

```java
public class DataEntryDTO implements Serializable {

    private int _timestep;
    @JsonGetter("timestep")
    public int getTimestep(){
        return this._timestep;
    }

    private float _vehicleCO;
    @JsonGetter("vehicle_CO")
    public float getVehicleCO(){
        return this._vehicleCO;
    }

    private float _vehicleCO2;
    @JsonGetter("vehicle_CO2")
    public float getVehicleCO2(){
        return this._vehicleCO2;
    }

    private float _vehicleHC;
    @JsonGetter("vehicle_HC")
    public float getVehicleHC(){
        return this._vehicleHC;
    }

    private float _vehicleNOx;
    @JsonGetter("vehicle_NOx")
    public float getVehicleNOx(){
        return this._vehicleNOx;
    }
```

```java
public class DataEntryDTO implements Serializable {

    private float _vehiclePMx;
    @JsonGetter("vehicle_PMx")
    public float getVehiclePMx(){
        return this._vehiclePMx;
    }

    private float _vehicleFuel;
    @JsonGetter("vehicle_fuel")
    public float getVehicleFuel(){
        return this._vehicleFuel;
    }

    private String _vehicleID;
    @JsonGetter("vehicle_id")
    public String getVehicleID(){
        return this._vehicleID;
    }

    private String _vehicleLane;
    @JsonGetter("vehicle_lane")
    public String getVehicleLane(){
        return this._vehicleLane;
    }

    private float _vehicleNoise;
    @JsonGetter("vehicle_noise")
    public float getVehicleNoise(){
        return this._vehicleNoise;
    }
```

```java
public class DataEntryDTO implements Serializable {

    private String _vehicleType;
    @JsonGetter("vehicle_type")
    public String getVehicleType(){
        return this._vehicleType;
    }

    private boolean _waiting;
    @JsonGetter("vehicle_waiting")
    public boolean getVehicleWaiting(){
        return this._waiting;
    }

    private float _vehicleX;
    @JsonGetter("vehicle_x")
    public float getVehicleX(){
        return this._vehicleX;
    }

    private float _vehicleY;
    @JsonGetter("vehicle_y")
    public float getVehicleY(){
        return this._vehicleY;
    }
```

Spark Structured Streaming

- Komande za pokretanje (nalazimo se u folderu projekta – "BigData-Projekat2")

docker build --rm -f spark-streaming-app ./SparkStreamingApp/

docker compose -f ./SparkStreaming/docker-compose.yml up -d --scale spark-worker=2

```
uros@uros-A5-X1:~/Master/BigData/Projekti/Projekat2$ docker compose -f ./SparkStreaming/docker-compose.yml up -d --scale spark-worker=2
[+] Running 9/13
 ⠿ Network mobility_infr                  Created                                                                                            64.9s
 ⠿ Volume "sparkstreaming_kafka_data"     Created                                                                                            64.8s
 ⠿ Volume "sparkstreaming_cassandra_data1" Created                                                                                           64.8s
 ⠿ Volume "sparkstreaming_zookeeper_data" Created                                                                                            64.8s
 ✔ Container cassandra-1                   Healthy                                                                                            63.3s
 ✔ Container spark                         Started                                                                                            0.8s
 ✔ Container sparkstreaming-spark-worker-2 Started                                                                                            1.1s
 ✔ Container sparkstreaming-spark-worker-1 Started                                                                                            0.5s
 ✔ Container zookeeper                     Started                                                                                            0.5s
 ✔ Container kafka                         Started                                                                                            1.0s
 ✔ Container init-cassandra                Exited                                                                                             63.5s
 ✔ Container consumer-app                  Started                                                                                            64.7s
 ✔ Container init-kafka                    Started                                                                                            1.4s
uros@uros-A5-X1:~/Master/BigData/Projekti/Projekat2$ docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED          STATUS                    PORTS                                                                      NAMES
053a5b31a78b   consumer-app             "python -u /consumer…"   About a minute ago   Up 9 seconds          9092/tcp, 0.0.0.0:9094->9094/tcp, :::9094->9094/tcp                        consumer-app
727101428c18   bitnami/kafka:latest     "/opt/bitnami/script…"   About a minute ago   Up About a minute     cassandra-1
821bdc4c57a8   cassandra:latest         "docker-entrypoint.s…"   About a minute ago   Up About a minute (healthy)  7000-7001/tcp, 7199/tcp, 9042/tcp, 9160/tcp                        cassandra-1
3b13ca19574f   bitnami/spark:latest     "/opt/bitnami/script…"   About a minute ago   Up About a minute     0.0.0.0:8080->8080/tcp, :::8080->8080/tcp                                  spark
7c5de96147a2   bitnami/spark:latest     "/opt/bitnami/script…"   About a minute ago   Up About a minute     sparkstreaming-spark-worker-1
8b17a9243491   bitnami/zookeeper:latest "/opt/bitnami/script…"   About a minute ago   Up About a minute     2888/tcp, 3888/tcp, 0.0.0.0:2181->2181/tcp, :::2181->2181/tcp, 8080/tcp    zookeeper
b230d5ac572e   bitnami/spark:latest     "/opt/bitnami/script…"   About a minute ago   Up About a minute     sparkstreaming-spark-worker-2
uros@uros-A5-X1:~/Master/BigData/Projekti/Projekat2$
```

Spark Structured Streaming

docker run -d --name spark-streaming-app -p 4040:4040 --network mobility_infr -e SPARK_STREAMING_APP_ARGS_WINDOW_SIZE=1_minute -e SPARK_STREAMING_APP_ARGS_WINDOW_SLIDE=20_seconds spark-streaming-app

- Napomena: Argumenti window_size I window_slide moraju da se navedu u formatu "duration"_"time_unit", kao na primeru iznad (npr. 1_minute, 5_minutes, 10_seconds, itd.)

java -jar ./SUMODataProducer/SUMODataProducer.jar ./SUMODataProducer/input traffic_data

- Pokrećemo KafkaProducer aplikaciju koja je "zapakovana" kao jar fajl. Potrebno je navesti putanju do input foldera I ime topica na koji se podaci šalju (kako je u obe streaming aplikacija ime topica "traffic_data", u data producer aplikaciji ne treba dirati ovaj parametar)

```
uros@uros-A5-X1:~/Master/BigData/Projekti/Projekat2$ java -jar ./SUMODataProducer/SUMODataProducer.jar ./SUMODataProducer/input traffic_data
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
New event. Event key(Vehicle ID) = bus0
New event. Event key(Vehicle ID) = motorcycle0
New event. Event key(Vehicle ID) = veh0
New event. Event key(Vehicle ID) = bus0
New event. Event key(Vehicle ID) = motorcycle0
New event. Event key(Vehicle ID) = veh0
New event. Event key(Vehicle ID) = veh1
New event. Event key(Vehicle ID) = bus0
New event. Event key(Vehicle ID) = motorcycle0
New event. Event key(Vehicle ID) = veh0
New event. Event key(Vehicle ID) = veh1
New event. Event key(Vehicle ID) = veh2
New event. Event key(Vehicle ID) = veh3
```

Rezultati

- Napomena: U Spark Streaming aplikaciji je moguće izostaviti argument window_slide i tada ćemo imati tumbling prozore veličine window_size
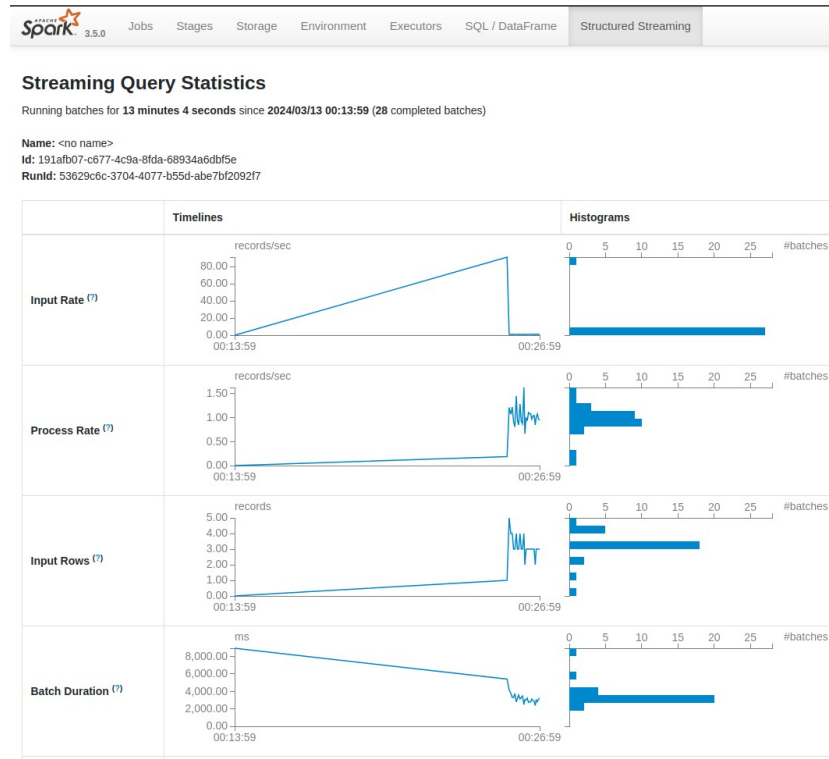
Rezultati

- Komandom docker run --rm -it --network mobility_infr nuvo/docker-cqlsh cqlsh cassandra-1 9042 --cqlversion=3.4.6 je moguće dobiti interaktivni shell za izvršavanje upita nad bazom u kojoj su smešteni rezultati obrade. Keyspace se zove *mobility*, ime familije kolona je *traffic_data*.

Flink Streaming

- Komande za pokretanje (nalazimo se u folderu projekta – "BigData-Projekat2")

  docker compose -f ./FlinkStreaming/docker-compose.yml up -d --scale taskmanager=2

Flink Streaming

- Aplikacija se submit-uje preko web-interfejsa na adresi "localhost:8081". Putanja do jar fajla aplikacije je
BigData-Projekat2/FlinkStreaming/FlinkStreamingApp/out/artifacts/FlinkStreamingApp_jar/
FlinkStreamingApp.jar
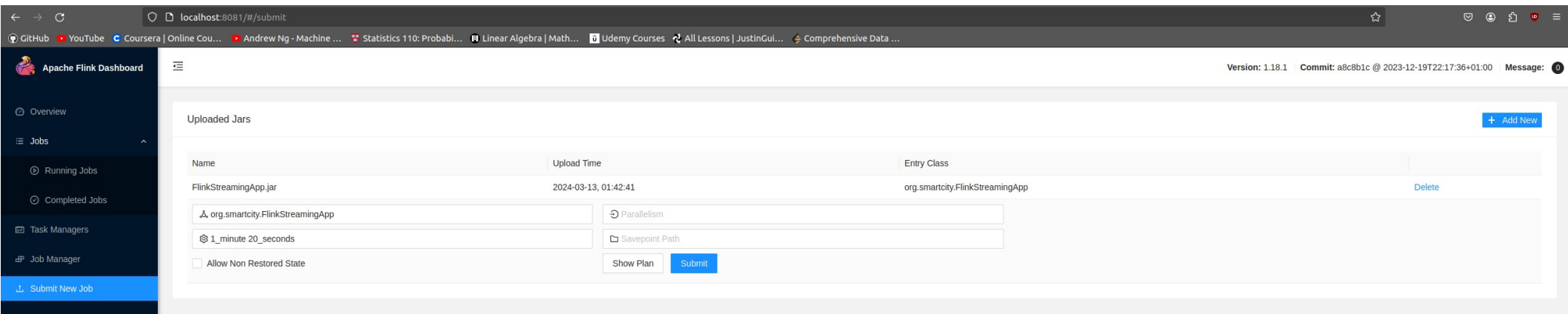
- Napomena: U Flink streaming aplikaciji je moguće navesti samo jedan argument, čime dobijamo tumbling prozore definisane veličine

Rezultati

- Nakon što pokrenemo streaming aplikaciju, istim komandama od malopre možemo pokrenuti DataProducer aplikaciju, a zatim vršiti upite nad bazom.

Rezultati