DALHOUSIE UNIVERSITY

Data Management, Warehousing and Analytics

Jaskaran Singh

MACS

B00948857

js356337@dal.ca

Lab Assignment 5

Gitlab Repository link:

https://git.cs.dal.ca/singh16/csci5408_s23_b00948857_jaskaran_singh.git

- Dataproc cluster has been created in GCP with name “cluster-5868-m”.

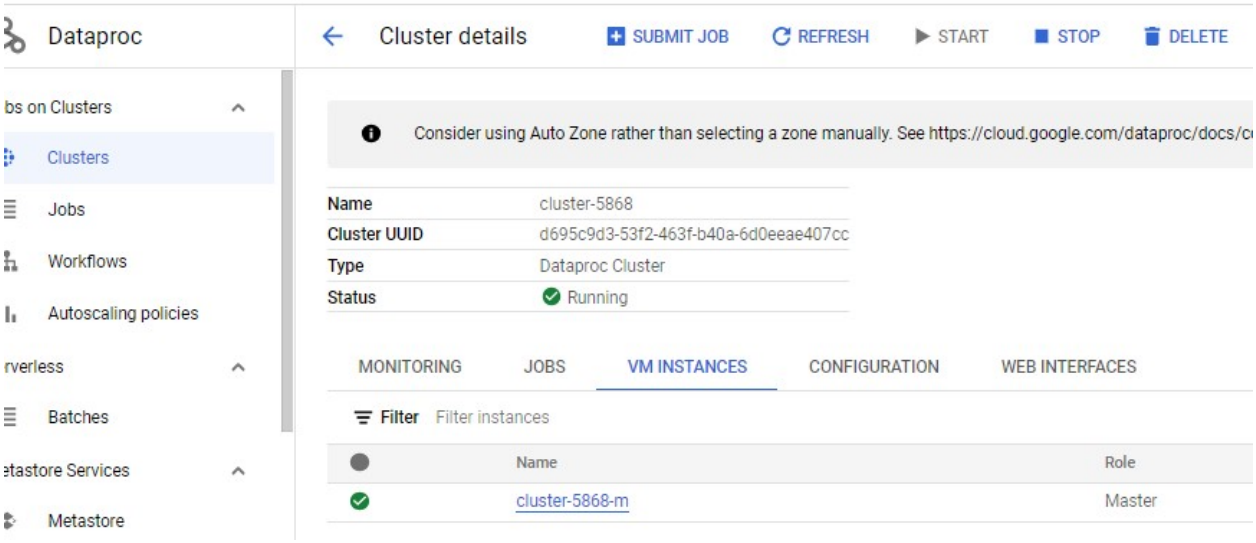

Figure: Cluster created in GCP

- A free account has been created on OpenWeather API and weather API has been hit but getting 401 response.

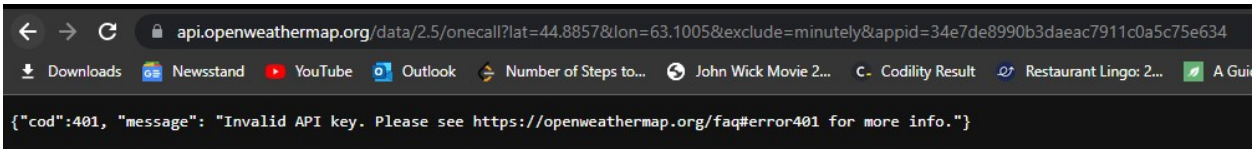

Figure: API giving 401 error

- Weather.json data has been downloaded from Teams app.
- A maven project has been created with Java 8 version.
- Dependencies for the same has been imported as is mentioned in pom.xml file.

```
<dependencies>
    <!-- https://mvnrepository.com/artifact/org.apache.spark/spark-sql -->
    <dependency>
        <groupId>org.apache.spark</groupId>
        <artifactId>spark-sql_2.13</artifactId>
        <version>3.4.1</version>
        <scope>provided</scope>
    </dependency>


    <!-- https://mvnrepository.com/artifact/org.apache.spark/spark-core -->
    <dependency>
        <groupId>org.apache.spark</groupId>
        <artifactId>spark-core_2.13</artifactId>
        <version>3.4.1</version>
    </dependency>
</dependencies>
```

Figure: Dependencies in pom.xml

- Using builder method of 'SparkSession', a SparkSession object is created.
- A dataset has been created after reading the json file with option ("multiline", **true**) using read method of SparkSession.

Defining the schema for nested struct:

- A StructType named feelsLikeSchema is defined to represent the schema of the "feels_like" nested struct.
- It contains four fields: "day", "night", "eve", and "morn" of DoubleType.

Filtering the data:

- The code selects specific columns from the weatherData DataFrame, including "lat", "lon", "timezone", "timezone_offset", and "daily".
- The explode() function is applied on the "daily" column to flatten the nested array of "daily" structs into individual rows.
- The .select() method is used to select specific columns from the exploded DataFrame.
- The "feels_like" column is casted to the defined schema using .withColumn() and col() functions.
- The DataFrame is filtered using .filter() to keep only the rows where "feels_like.day" is greater than 15.

Excluding unnecessary fields:

- The unnecessary fields are dropped from the filtered DataFrame using .drop() method.
- The fields include "feels_like.night", "feels_like.eve", "feels_like.morn", "pressure", "humidity", "dew_point", "wind_speed", "wind_deg", "wind_gust", "weather", "clouds", "pop", "rain", and "uvi".

Saving the filtered data:

- The filtered DataFrame is saved to a new file named "summer_weather.json" using .write().json() method.

Stopping the SparkSession:

- Finally, the SparkSession is stopped using spark.stop() to release the resources.

Building the Maven project:

- After this, build the project using mvn IntelliJ Maven Lifecycles.
- Next, copy the jar file and weather.json file and upload them in cloud shell on GCP.
- Run spark-submit command and also use flag - - class to give path to class ith main method.

```
singhjaskaran762@cluster-5868-m:~$ spark-submit --class org.example.SparkApp SparkApplication-1.0-SNAPSHOT.jar
```

- After successful execution, a new json file named “summer_weather.json” will be created.

```
singhjaskaran762@cluster-5868-m:~$ ls
SparkApplication-1.0-SNAPSHOT.jar  summer_weather.json  weather.json
singhjaskaran762@cluster-5868-m:~$ cat summer_weather.json
[
  {
    "lat": 44.6462,
    "lon": -63.5736,
    "timezone": "America/Halifax",
    "timezone_offset": -10800,
    "dt": 1655913600,
    "sunrise": 1655886551,
    "sunset": 1655942600,
    "day": 17.74
  },
  {
    "lat": 44.6462,
    "lon": -63.5736,
    "timezone": "America/Halifax",
    "timezone_offset": -10800,
    "dt": 1656000000,
    "sunrise": 1655972966,
    "sunset": 1656029009,
    "day": 20.17
  },
  {
    "lat": 44.6462,
    "lon": -63.5736,
    "timezone": "America/Halifax",
    "timezone_offset": -10800,
    "dt": 1656086400,
    "sunrise": 1656059384,
    "sunset": 1656115415,
    "day": 16.27
  },
  {
    "lat": 44.6462,
    "lon": -63.5736,
    "timezone": "America/Halifax",
    "timezone_offset": -10800,
```

Figure: Commands on GCP Shell

References:

- Apache Spark
  https://spark.apache.org/docs/latest/sql-data-sources-json.html
- Spark tutorial, GCP
  https://cloud.google.com/dataproc/docs/tutorials/gcs-connector-spark-tutorial#before-you-begin
- Analytics Vidhya
  https://www.analyticsvidhya.com/blog/2021/09/beginners-guide-to-create-pyspark-dataframe/