Flink Project

Yellow Taxi Trip Data

Description: New York Taxi and Limousine Commission stores all trips done by yellow and green taxis. This data is reported by each taxi and is sent to a data center that processes this information in order to detect irregular situations.

Each taxi reports all the information regarding each trip with the following format: VendorID, tpep_pickup_datetime, tpep_dropoff_datetime, passenger_count, trip_distance, RatecodeID, store_and_fwd_flag, PULocationID, DOLocationID, payment_type, fare_amount, extra, mta_tax, tip_amount, tolls_amount, improvement_surcharge, total_amount, congestion_surcharge.

VendorID: A code indicating the TPEP provider that provided the record. 1 = Creative Mobile Technologies, LLC; 2 = VeriFone Inc…

tpep_pickup_datetime: The date and time when the meter was engaged.

tpep_dropoff_datetime: The date and time when the meter was disengaged.

passenger_count: The number of passengers in the vehicle. This is a driver-entered value.

trip_distance: The elapsed trip distance in miles reported by the taximeter.

RatecodeID: The final rate code in effect at the end of the trip. 1 = Standard rate; 2 = JFK; 3 = Newark; 4 = Nassau or Westchester; 5 = Negotiated fare; 6 = Group ride.

store_and_fwd_flag: This flag indicates whether the trip record was held in vehicle memory before sending it to the vendor, aka “store and forward,” because the vehicle did not have a connection to the server. Y= store and forward trip; N= not a store and forward trip.

PULocationID: TLC Taxi Zone in which the taximeter was engaged.

DOLocationID: TLC Taxi Zone in which the taximeter was disengaged.

payment_type: A numeric code signifying how the passenger paid for the trip. 1 = Credit card; 2 = Cash; 3 = No charge; 4 = Dispute; 5 = Unknown; 6 = Voided trip.

fare_amount: The time-and-distance fare calculated by the meter.

Extra: Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.

mta_tax: $0.50 MTA tax that is automatically triggered based on the metered rate in use.

tip_amount: Tip amount – This field is automatically populated for credit card tips. Cash tips are not included.

tolls_amount: Total amount of all tolls paid in trip.

improvement_surcharge: $0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.

total_amount: The total amount charged to passengers. It does not include cash tips.

congestion_surcharge: The surcharge applied when the trips goes through a congested area.

The goal of this project is to **develop a Java program using *Flink* for implementing the following functionality:**

1. **JFK airport trips.** It informs about the trips ending at JFK airport with two or more passengers each hour for each vendorID. The output must have the following format: vendorID, tpep_pickup_datetime, tpep_dropoff_datetime, passenger_count. Being tpep_dropoff_datetime the time the last trip finishes, tpep_pickup_datetime the starting time of the first trip and passenger_count the total number of passengers.

2. **Large trips.** It reports the vendors that do 5 or more trips during 3 hours that take at least 20 minutes. The output has the following format: VendorID, day, numberOfTrips, tpep_pickup_datetime, tpep_dropoff_datetime, being tpep_dropoff_datetime the time the last trip finishes and tpep_pickup_datetime the starting time of the first trip.

Notes:

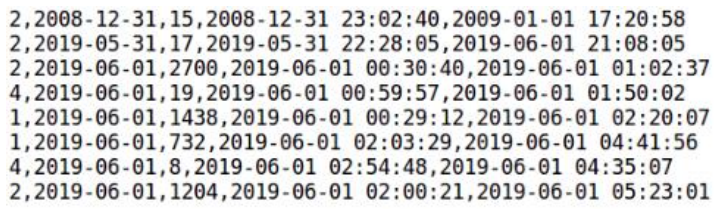
- The hours are defined based on the tpep_pickup_datetime field.
- JFK airport trips output example:

```
2,2008-12-31 23:11:48,2009-01-01 01:45:44,2
2,2009-01-01 00:02:55,2009-01-01 17:20:58,2
2,2019-05-31 23:43:38,2019-05-31 23:51:09,8
2,2019-06-01 00:20:06,2019-06-01 00:32:46,68
1,2019-06-01 00:24:26,2019-06-01 01:37:35,28
1,2019-06-01 01:18:33,2019-06-01 02:07:02,22
2,2019-06-01 01:15:18,2019-06-01 02:24:18,40
2,2019-06-01 02:07:28,2019-06-01 02:56:42,14
```

Each row has this format:

vendorID, tpep_pickup_datetime, tpep_dropoff_datetime, passenger_count

- Large trips output example:



```
2,2008-12-31,15,2008-12-31 23:02:40,2009-01-01 17:20:58
2,2019-05-31,17,2019-05-31 22:28:05,2019-06-01 21:08:05
2,2019-06-01,2700,2019-06-01 00:30:40,2019-06-01 01:02:37
4,2019-06-01,19,2019-06-01 00:59:57,2019-06-01 01:50:02
1,2019-06-01,1438,2019-06-01 00:29:12,2019-06-01 02:20:07
1,2019-06-01,732,2019-06-01 02:03:29,2019-06-01 04:41:56
4,2019-06-01,8,2019-06-01 02:54:48,2019-06-01 04:35:07
2,2019-06-01,1204,2019-06-01 02:00:21,2019-06-01 05:23:01
```

Each row has this format:
VendorID, day, numberOfTrips, tpep_pickup_datetime, tpep_dropoff_datetime, being tpep_dropoff_datetime

Input: The Java program will read the events from a CSV with the information the taxis report.

A sample file is available at https://dl.lsdupm.ovh/yellow_tripdata_2019_06.csv

Output to be generated:

The program must generate 2 output CSV files.

- jfkAlarms.csv: stores the output of the *JFS airport trips.*
  - format: vendorID, tpep_pickup_datetime, tpep_dropoff_datetime, passenger_count
- largeTrips.csv: stores the output of the *Large trips.*
  - format: VendorID, day, numberOfTrips.

Requirements:

The application must be developed using these versions of the software: Oracle Java 8, Flink 1.9.1. The application will be deployed using Ubuntu.

The parallelism for the write operation to the files must be 1.

The main class of the project must be named **master2019.flink.YellowTaxiTrip**, the application will be tested using the following procedure from the root folder of your project:

- mvn clean package
- flink run -c master2019.flink.YellowTaxiTrip target/$YOUR_JAR_FILE --input $PATH_TO_INPUT_FILE --output $PATH_TO_OUTPUT_FOLDER

The input file and the output folder will exist on all nodes of the cluster running the *Flink* Task Managers.

The project must use the skeleton available at: https://dl.lsdupm.ovh/YellowTaxiTrip.zip

Submission:

- **Deadline:** 12th January 2020 at 23:55
- **Where: All the required files must be uploaded to Moodle by the deadline.** The file must be named **surname1-surname2.zip**. This file has a project named *yellowTaxiTrip* and with the pom file and the source code in the folder src/ which corresponds to the provided skeleton:
    - Surname1-surname2.zip
        - yellowTaxiTrip
            - pom.xml
            - src/
- **Groups:** The project is implemented by 2 persons from the same master program.