Προχωρημένα Θέματα Βάσεων Δεδομένων

Εξαμηνιαία Εργασία

Ονοματεπώνυμο: Παπανικολάου Γρηγόρης

Αριθμός Μητρώου: el18649

Github: https://github.com/gr3gor1/Adv-DBs

Ερώτηση 1

Σχετικά με την εγκατάσταση της πλατφόρμας εκτέλεσης Spark & HDFS χρησιμοποιήσαμε δύο μηχανήματα μέσω Okeanos, τα οποία έχουν τις ακόλουθες προδιαγραφές:

- 4 cores (CPU)
- 8 GB (RAM)
- 30 GB

Τα δύο μηχανήματα λειτουργούν ως datanodes και το ένα εξ' αυτών λειτουργεί και ως namenode. Θα ξεκινήσουμε, κάνοντας τις απαραίτητες αλλαγές στο /etc/hosts σε κάθε μηχάνημα, έτσι ώστε να είναι γνωστές οι IPv4 διευθύνσεις που απαιτούνται και θα δώσουμε την ικανότητα στο namenode να συνδέεται με τους datanodes με ssh χωρίς να υπάρχει η ανάγκη password.

Στη συνέχεια, αναφορικά με τη διαδικασία εγκατάστασης, θα συμπληρώσουμε στο .bashrc του κάθε μηχανήματος τα απαιτούμενα paths και θα προχωρήσουμε στη συμπλήρωση των configuration files σε κάθε μηχάνημα για την λειτουργία του HDFS και του YARN καθώς και στην εγκατάσταση των υπόλοιπων εργαλείων που χρειαζόμαστε (Spark,Python3.8,Java). Ενδεικτικά, όσον αφορά τη διαδικασία της εγκατάστασης, για να πετύχουμε τη λειτουργία namenode - datanode στον master, φροντίζουμε οι πληροφορίες των αντίστοιχων daemons, ν' ανατίθενται σε διαφορετικά directories, τροποποιώντας κατάλληλα το hdfs-site.xml. Ακόμα στο hdfs-site.xml θέτουμε τιμή dfs.replication ίση με 3, έτσι ώστε να έχουμε τρία αντίγραφα των επιμέρους τμημάτων αρχείων, κατανεμημένα στα μηχανήματα με λειτουργία datanode.

Έχουμε δημιουργήσει τ' ακόλουθα directories στο HDFS:

- /data/csv
- /data/parq

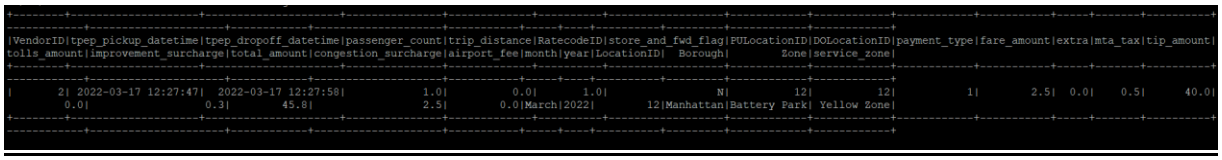
Στη συνέχεια, θα παραθέσουμε τον κώδικα, με τον οποίο μπορούμε να δημιουργήσουμε τ' απαραίτητα dataframes και RDDs με χρήση των δεδομένων, από τα παραπάνω directories.

```
from pyspark.sql.functions import *
from pyspark.sql.types import *
from pyspark.sql import SparkSession
import os
import sys

os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable

spark = SparkSession.builder.master("spark://192.168.0.1:7077").appName("DFs-RDDs").getOrCreate()

#read all parquets into a single dataframe containing the trips data
trips=spark.read.parquet("hdfs://192.168.0.1:9000/data/parq")

#read the csv file from the directory we created in HDFS and turn it into a dataframe
zones=spark.read.option("header","true").option("delimiter",",").option("inferSchema","true").csv("hdfs://192.168.0.1:9000/data/csv/taxi+_zone_lookup.csv")

#now all we have to do is turn our dataframes to RDDs
rddtrips = trips.rdd
rddzones = zones.rdd

#then we can check if everything is okay
trips.show()
zones.show()

a = rddtrips.take(10)
print(a)

b = rddzones.take(10)
print(b)
```

Ερώτηση 2

Q1

Στο συγκεκριμένο ερώτημα, κρατήσαμε μόνο εγγραφές που αφορούν τον Μάρτιο του 2022 και στη συνέχεια κάνοντας join με το dataframe, που έχει τα αναγνωριστικά και τα ονόματα των επιμέρους περιοχών, βρήκαμε το μέγιστο tip amount των διαδρομών, που καταλήγουν στο Battery Park. Υπάρχει ωστόσο, άλλη μια καταχώρηση με όνομα Battery Park City, στις αντιστοιχίσεις των αναγνωριστικών, την οποία δεν θεωρούμε κομμάτι του ερωτήματος.



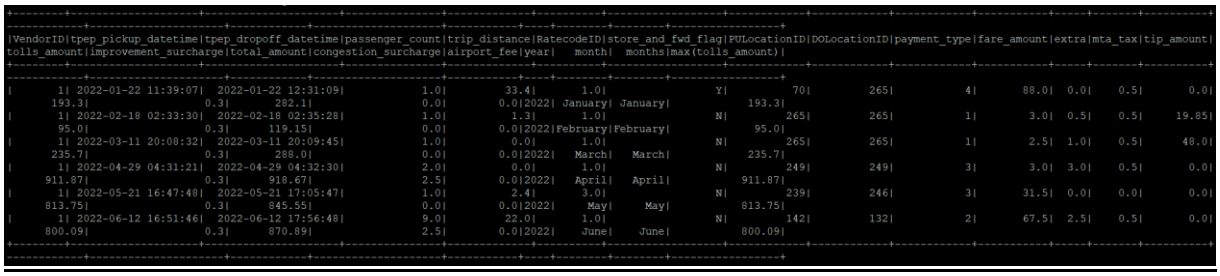
```
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----------+-----+-------+----------+
|VendorID|tpep_pickup_datetime|tpep_dropoff_datetime|passenger_count|trip_distance|RatecodeID|store_and_fwd_flag|PULocationID|DOLocationID|payment_type|fare_amount|extra|mta_tax|tip_amount|
tolls_amount|improvement_surcharge|total_amount|congestion_surcharge|airport_fee|month|year|LocationID|  Borough|        Zone|service_zone|
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----------+-----+-------+----------+
-----------+--------------------+------------+--------------------+-----------+-----+---+----------+---------+------------+------------+
|       2| 2022-03-17 12:27:47| 2022-03-17 12:27:58|            1.0|          0.0|       1.0|                 N|          12|          12|           1|        2.5|  0.0|    0.5|      40.0|
       0.0|                 0.3|        45.8|                2.5|        0.0|March|2022|        12|Manhattan|Battery Park| Yellow Zone|
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----------+-----+-------+----------+
-----------+--------------------+------------+--------------------+-----------+-----+---+----------+---------+------------+------------+
```

Χρόνος μ' έναν worker: 56 sec

Χρόνος με δύο workers: 33 sec

Q2

Στο συγκεκριμένο ερώτημα, αφού κρατήσαμε μόνο εγγραφές για τη χρονιά και τους μήνες που μας ενδιαφέρουν, εντοπίσαμε ανά μήνα το μέγιστο ποσό που δόθηκε σε διόδια. Στη συνέχεια, κάνοντας join το αποτέλεσμα, με το αρχικό dataframe και κρατώντας μόνο εγγραφές, που ταιριάζουν τα μεγέθη μεταξύ τους (months == month && tolls_amount==max(tolls_amount)), έχουμε τα ζητούμενα αποτελέσματα.



```
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----------+-----+-------+----------+
|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|airport_fee|year|   month|  months|max(tolls_amount)|
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----------+-----+-------+----------+
-----------+--------------------+------------+--------------------+-----------+----+--------+--------+-----------------+
|       1| 2022-01-22 11:39:07| 2022-01-22 12:31:09|            1.0|         33.4|       1.0|                 Y|          70|         265|           4|       88.0|  0.0|    0.5|       0.0|
      193.3|                 0.3|       282.1|                0.0|        0.0|2022| January| January|            193.3|
|       1| 2022-02-18 02:33:30| 2022-02-18 02:35:28|            1.0|          1.3|       1.0|                 N|         265|         265|           1|        3.0|  0.5|    0.5|     19.85|
       95.0|                 0.3|       119.15|                0.0|        0.0|2022|February|February|             95.0|
|       1| 2022-03-11 20:08:32| 2022-03-11 20:09:45|            1.0|          0.0|       1.0|                 N|         265|         265|           1|        2.5|  1.0|    0.5|      48.0|
     235.7|                 0.3|       288.0|                0.0|        0.0|2022|   March|   March|            235.7|
|       1| 2022-04-29 04:31:21| 2022-04-29 04:32:30|            2.0|          0.0|       1.0|                 N|         249|         249|           3|        3.0|  3.0|    0.5|       0.0|
     911.87|                 0.3|       918.67|                2.5|        0.0|2022|   April|   April|            911.87|
|       1| 2022-05-21 16:47:48| 2022-05-21 17:05:47|            1.0|          2.4|       3.0|                 N|         239|         246|           3|       31.5|  0.0|    0.0|       0.0|
     813.75|                 0.3|       845.55|                0.0|        0.0|2022|     May|     May|            813.75|
|       1| 2022-06-12 16:51:46| 2022-06-12 17:56:48|            9.0|         22.0|       1.0|                 N|         142|         132|           2|       67.5|  2.5|    0.5|       0.0|
     800.09|                 0.3|       870.89|                2.5|        0.0|2022|    June|    June|            800.09|
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----------+-----+-------+----------+
-----------+--------------------+------------+--------------------+-----------+----+--------+--------+-----------------+
```

Χρόνος μ' έναν worker: 44 sec

Χρόνος με δύο workers: 24 sec

Ερώτηση 3

Q3

Στο συγκεκριμένο ερώτημα, έχοντας κρατήσει μόνο εγγραφές που αφορούν το διάστημα των μηνών που μας ενδιαφέρουν το 2022, με βάση τη στήλη day of the year κάνουμε ομαδοποίηση ανά δεκαπενθήμερο (η μέτρηση ξεκινά από το 1 οπότε η πρώτη περίοδος περιέχει 14 ημέρες). Ανά περίοδο, υπολογίζουμε την μέση απόσταση και το μέσο συνολικό κόστος των διαδρομών, που ανήκουν σ' αυτή.

15_day_period	avg(Trip_distance)	avg(Total_amount)
0	5.375900812549739	20.028410224160325
1	4.955629059731223	18.954197049834928
2	5.968934443193151	19.5560476912666
3	6.31243736438367	20.11378437270254
4	6.480485434052824	20.652278174179074
5	5.613652667238764	21.108061236787446
6	5.649882510489324	21.49729946083937
7	5.813096714425007	21.476737125703 15
8	6.080568966454786	21.786113851955296
9	7.999029204040286	22.793742812773953
10	6.436759370743996	22.497024880568375
11	6.166598205960558	22.381115337159258
12	5.9328103987614265	22.019707932064964

Χρόνος μ' έναν worker: 31 sec

Χρόνος με δύο workers: 28 sec

Q6 (RDD)

Δημιουργούμε tuples τριών στοιχείων, τα οποία περιέχουν αρχικά την τιμή trip_distance στη συνέχεια την τιμή total amount και τέλος την τιμή (day of year // 15). Στη συνέχεια θέτουμε ως κλειδί, τον αριθμό της περιόδου, που αφορά το κάθε tuple και υπολογίζουμε τον μέσο όρο των άλλων στηλών, χρησιμοποιώντας reduce και έναν counter, για να πραγματοποιήσουμε την τελική διαίρεση.

```
[(0, 5.375900812549739, 20.028410224160325), (1, 4.955629059731223, 18.954197049834928), (2, 5.968934443193151, 19.5560476912666), (3, 6.31243736438367, 20.11378437270254), (4, 6.480485434052824, 20.652278174179074), (5, 5.613652667238764, 21.108061236787446), (6, 5.649882510489324, 21.49729946083937), (7, 5.813096714425007, 21.476737125703 15), (8, 6.080568966454786, 21.786113851955296), (9, 7.999029204040286, 22.793742812773953), (10, 6.436759370743996, 22.497024880568375), (11, 6.166598205960558, 22.381115337159258), (12, 5.9328103987614265, 22.019707932064964)]
```

Χρόνος μ' έναν worker: 6,2 min

Χρόνος με δύο workers: 1,8 min

Ερώτηση 4

Q4

Κρατώντας μόνο τη χρονική περίοδο που μας αφορά, βρίσκουμε ανά ημέρα της εβδομάδας και ανά ώρα (η ένδειξη ώρας σηματοδοτεί την έναρξη π.χ. 22 => 22-23) το συνολικό αριθμό ανθρώπων, που βρίσκονται σε ταξί. Στη συνέχεια, με χρήση ενός window (partioning by day) ανά ημέρα και με βάση φθίνουσα ταξινόμηση των συνόλων, δημιουργούμε εφτά επιμέρους indexes (τα οποία στη συνέχεια διαγράφουμε), για να έχουμε τις τρεις μεγαλύτερες ώρες αιχμής ανά ημέρα.

day	hour	sum(Passenger_count)
Friday	22	255878.0
Friday	20	282941.0
Friday	21	289408.0
Monday	20	247418.0
Monday	19	236534.0
Monday	21	238259.0
Saturday	20	272951.0
Saturday	19	261720.0
Saturday	21	274010.0
Sunday	17	226426.0
Sunday	19	226543.0
Sunday	24	228580.0
Thursday	21	283074.0
Thursday	19	268112.0
Thursday	20	285365.0
Tuesday	21	268951.0
Tuesday	19	257625.0
Tuesday	20	276200.0
Wednesday	20	281426.0
Wednesday	21	276147.0
Wednesday	19	258958.0

Χρόνος μ' έναν worker: 37 sec

Χρόνος με δύο workers: 29 sec

Q5

Κρατώντας μόνο εγγραφές στην περίοδο που μας ενδιαφέρει, προσθέτουμε μια επιπλέον στήλη, έτσι ώστε σε κάθε διαδρομή, να εμφανίζεται και το ζητούμενο ποσοστό. Στη συνέχεια ανά μήνα, ημέρα του μήνα και ημέρα της εβδομάδας, βρίσκουμε το μέσο όρο των ποσοστών. Έπειτα δημιουργούμε ένα index, με βάση ένα window (όπως στο Q4) που αφορά partioning σε μήνες και κρατάμε τις πέντε καλύτερες ημέρες.

month	day_of_month	day	avg(percentage)
January	9	Sunday	45.78674775487207
January	31	Monday	43.93563580770273
January	1	Saturday	29.07803686136836
January	29	Saturday	24.059518454370057
January	16	Sunday	23.377299918220096
February	21	Monday	25.981657452766274
February	13	Sunday	24.572068389402546
February	9	Wednesday	23.904535643412483
February	10	Thursday	23.33961589934868
February	27	Sunday	23.3006799515465
March	18	Friday	29.671341612659685
March	21	Monday	27.57992602492248
March	26	Saturday	22.70884595372165
March	5	Saturday	22.55546137249565
March	12	Saturday	22.100859110808635
April	12	Tuesday	48.36884410450339
April	2	Saturday	31.175092883998968
April	21	Thursday	30.44861250236277
April	3	Sunday	24.46372770475391
April	30	Saturday	21.99676965994668
May	12	Thursday	32.402658973198044
May	20	Friday	26.034036090366385
May	16	Monday	23.659110789279985
May	15	Sunday	22.05244524700949
May	6	Friday	21.832006161884486
June	13	Monday	38.45136993724611
June	25	Saturday	32.91307329265353
June	10	Friday	27.397637812780694
June	16	Thursday	25.534975757875227
June	20	Monday	24.242914593519107

Χρόνος μ' έναν worker: 38 sec

Χρόνος με δύο workers: 31 sec

Σημείωση: Ορισμένα στιγμιότυπα δεν φαίνονται επαρκώς ωστόσο οι λεπτομέρειες είναι εμφανείς αν γίνει zoom-in

<u>Παρακάτω παρέχεται το στιγμιότυπο των αποτελεσμάτων για κάθε query με δύο workers και έναν worker.</u>

Τελικοί χρόνοι με δύο workers :

app-20230204131724-0011	Q6	8	512.0 MiB		2023/02/04 13:17:24	user	FINISHED	1.8 min
app-20230204131633-0010	Q5	8	512.0 MiB		2023/02/04 13:16:33	user	FINISHED	31 s
app-20230204131328-0009	Q4	8	512.0 MiB		2023/02/04 13:13:28	user	FINISHED	29 s
app-20230204131225-0008	Q3	8	512.0 MiB		2023/02/04 13:12:25	user	FINISHED	28 s
app-20230204131108-0007	Q2	8	512.0 MiB		2023/02/04 13:11:08	user	FINISHED	24 s
app-20230204125332-0006	Q1	8	512.0 MiB		2023/02/04 12:53:32	user	FINISHED	33 s

Τελικοί χρόνοι μ’ έναν worker :

app-20230204134024-0017	Q6	4	512.0 MiB		2023/02/04 13:40:24	user	FINISHED	6.2 min
app-20230204133854-0016	Q5	4	512.0 MiB		2023/02/04 13:38:54	user	FINISHED	38 s
app-20230204133719-0015	Q4	4	512.0 MiB		2023/02/04 13:37:19	user	FINISHED	37 s
app-20230204133639-0014	Q3	4	512.0 MiB		2023/02/04 13:36:39	user	FINISHED	31 s
app-20230204133232-0013	Q2	4	512.0 MiB		2023/02/04 13:32:32	user	FINISHED	44 s
app-20230204132816-0012	Q1	4	512.0 MiB		2023/02/04 13:28:16	user	FINISHED	56 s