Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Ακ. έτος 2022-2023, 9ο Εξάμηνο:Προχωρημένα Θέματα Βάσεων Δεδομένων

Ξυλιά Παναγιώτα Μικαέλα AM:03118859

Σάλιαρη Νικολέτα AM:03118864

ομάδα-55

github repo: https://github.com/mikaelaksil/atds-ntua-2023

hadoop UI:http://83.212.81.56:50070/

spark UI:http://83.212.81.56:8080/

Γραπτή αναφορά εξαμηνιαίας εργασίας

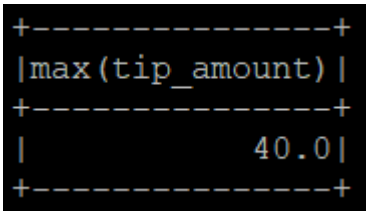
Στην εργασία κληθήκαμε να επεξεργαστούμε δεδομένα που αφορούν στις διαδρομές ταξί στην πόλη της Νέας Υόρκης.Χρησιμοποιήθηκε η πλατφόρμα Apache Spark και ως σύστημα αποθήκευσης αρχείων χρησιμοποιήθηκε HDFS. Χρησιμοποιήθηκαν VM - linux servers μεσω του Okeanos.

Για την υλοποίηση των queries Q1-Q5 επιλέξαμε γλώσσα Python.

Για την εκτέλεση των queries εκτελουμε την εντολή python3.8 q1.py στο VM, αντίστοιχα q2.py, q3.py, q4.py, q5.py, q3-rdd.py.

Παρακάτω παραθέτουμε τα αποτελέσματα για τα Q1, Q2, Q3, Q4, Q5 καθώς και τους χρόνους εκτέλεσης σε Dataframe API/ SQL API:

Q1:Να βρεθεί η διαδρομή με το μεγαλύτερο φιλοδώρημα (tip) τον Μάρτιο και σημείο άφιξης το "Battery Park".

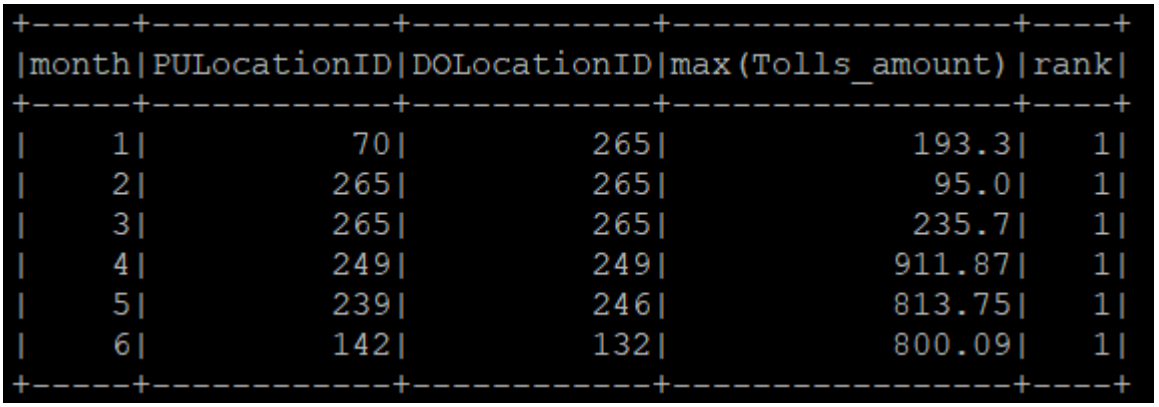


```
+---------------+
|max(tip_amount)|
+---------------+
|           40.0|
+---------------+
```

χρόνος εκτέλεσης για 1 worker/vm: 3.481260299682617 sec

χρόνος εκτέλεσης για 2 workers/vm: 3.3182358741760254 sec

Q2:Να βρεθεί, για κάθε μήνα, η διαδρομή με το υψηλότερο ποσό στα διόδια. Αγνοήστε μηδενικά ποσά.



```
+-----+------------+------------+------------------+----+
|month|PULocationID|DOLocationID|max(Tolls_amount)|rank|
+-----+------------+------------+------------------+----+
|    1|          70|         265|             193.3|   1|
|    2|         265|         265|              95.0|   1|
|    3|         265|         265|             235.7|   1|
|    4|         249|         249|            911.87|   1|
|    5|         239|         246|            813.75|   1|
|    6|         142|         132|            800.09|   1|
+-----+------------+------------+------------------+----+
```

χρόνος εκτέλεσης για 1 worker/vm: 15.730663299560547 sec

χρόνος εκτέλεσης για 2 workers/vm: 16.382025957107544 sec

Q3:Να βρεθεί, ανά 15 ημέρες, ο μέσος όρος της απόστασης και του κόστους για όλες τις διαδρομές με σημείο αναχώρησης διαφορετικό από το σημείο άφιξης.

15day_interval	avg_distance	avg_cost
2022-06-28 00:00:00	6.860061550499997	21.874996087497827
2022-02-14 00:00:00	5.760350176330592	19.652856555784588
2022-03-07 00:00:00	9.514659704294784	21.2115425316427
2022-04-05 00:00:00	3.563945007748453	21.06712402037446
2022-01-21 00:00:00	5.002730008921917	18.594950176838324
2022-05-23 00:00:00	8.411345709247614	23.28708904781351
2022-05-31 00:00:00	6.482740284739536	22.239324172378055
2022-05-14 00:00:00	6.006645974509848	21.455595950603822
2022-04-01 00:00:00	4.899253615682089	21.53802606296694
2022-01-15 00:00:00	7.9178225959261255	18.447453576514892
2022-05-11 00:00:00	7.312835513191856	21.632623759485103
2022-01-02 00:00:00	6.356628666535441	22.675210730721197
2022-03-29 00:00:00	4.4179480088495 14	20.810982933003633
2022-03-24 00:00:00	3.482177242606271	20.935335585184593
2022-01-29 00:00:00	16.501672401354078	17.403628947219303
2022-05-16 00:00:00	8.331618123884585	22.66804408563568
2022-05-09 00:00:00	9.560707671756857	22.482491777743434
2022-06-05 00:00:00	8.218535386443346	22.737683064724763
2022-04-09 00:00:00	6.81803265445999	20.67089436665914
2022-04-18 00:00:00	7.0699134341554775	21.789258124558536

only showing top 20 rows

χρόνος εκτέλεσης για 1 worker/vm: 14.996756553649902 sec

χρόνος εκτέλεσης για 2 workers/vm: 15.119275331497192 sec

Q4:Να βρεθούν οι τρεις μεγαλύτερες (top 3) ώρες αιχμής ανά ημέρα της εβδομάδος, εννοώντας τις ώρες (π.χ., 7-8πμ, 3-4μμ, κλπ) της ημέρας με τον μεγαλύτερο αριθμό επιβατών σε μια κούρσα ταξί. Ο υπολογισμός αφορά όλους τους μήνες.

day_of_week	pickup_hour	sum(Passenger_count)	rank
1	17	226433.0	3
1	0	228582.0	1
1	19	226550.0	2
2	21	238260.0	2
2	20	247418.0	1
2	19	236539.0	3
3	20	276202.0	1
3	21	268956.0	2
3	19	257630.0	3
4	20	281426.0	1
4	21	276152.0	2
4	19	258967.0	3
5	20	285368.0	1
5	21	283080.0	2
5	19	268121.0	3
6	20	282944.0	2
6	22	255879.0	3
6	21	289410.0	1
7	21	274011.0	1
7	20	272954.0	2

only showing top 20 rows

χρόνος εκτέλεσης για 1 worker/vm: 18.02756667137146 sec

χρόνος εκτέλεσης για 2 workers/vm: 18.330271244049072 sec

Q5:Να βρεθούν οι κορυφαίες πέντε (top 5) ημέρες ανά μήνα στις οποίες οι κούρσες είχαν το μεγαλύτερο ποσοστό σε tip. Για παράδειγμα, εάν η κούρσα κόστισε 10$ (fare_amount) και το tip ήταν 5$, το ποσοστό είναι 50%.

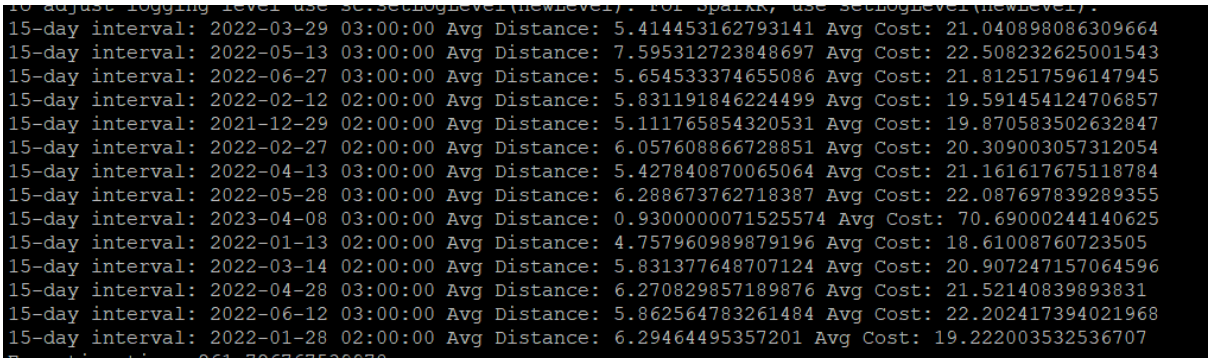
day_of_month	month	avg(tip_percentage)	rank
2	1	80.6934776619449	1
4	1	79.84822745776707	2
5	1	79.6097385007352	3
10	1	79.22094909674422	4
6	1	79.2117705713473	5
28	2	79.07517699288532	1
2	2	79.0698423559008	2
20	2	79.02337184266338	4
1	2	79.05946141020189	3
7	2	78.94896370247994	5
14	3	79.31302792507995	2
20	3	79.46958288509721	1
27	3	79.12810296306988	3
28	3	79.03968217152354	4
25	3	79.02322358052356	5
23	4	79.60285259298128	5
11	4	79.82650068697407	1
17	4	79.8177265178778	2
18	4	79.72870556444194	3
16	4	79.67223624140443	4

only showing top 20 rows

χρόνος εκτέλεσης για 1 worker/vm: 24.264857053756714 sec

χρόνος εκτέλεσης για 2 workers/vm: 24.450012922286987 sec

Τέλος μας ζητήθηκε να υλοποιήσουμε το Q3 και σε RDD-API, παρατηρούμε τα παρακάτω αποτελέσματα, καθώς και τον χρόνο εκτέλεσης:



χρόνος εκτέλεσης για 1 worker/vm: 259.6203727722168 sec

χρόνος εκτέλεσης για 2 workers/vm: 261.726767539978 sec

Σχολιασμός:

Σύγκριση RDD-API με SQL-API/Dataframe API

Παρατηρούμε πως το RDD-API κάνει εμφανώς περισσότερο χρόνο από ότι το SQL-API αυτό οφείλεται στο ότι τα RDDS χρειάζονται περισσότερο χρόνο για να εκτελέσουν απλά functions που αφορούν στο γκρουπάρισμα των δεδομένων.Επίσης, το Dataframe/SQL API διαθέτει schema ,οπότε μπορεί να έχει εξτρα πληροφορίες για τον τύπο των δεδομένων και τη δομή τους ,κάτι που δεν συμβαίνει στην περίπτωση των RDDs.Τέλος, το Dataframe/SQL API κάνει χρήση ενσωματωμένου optimizer, οπότε το query processing καθίσταται πιο εύκολο και πιο γρήγορο,τα δεδομένα δηλαδή γκρουπάρονται πιο γρήγορα και τα functions που εφαρμόζονται πάνω τους “τρέχουν” πιο γρήγορα.

Σύγκριση 1 worker - 2 workers:

Παρόλο που ίσως περιμέναμε η χρήση των 2 workers να επιφέρει πιο σύντομα το αποτέλεσμα του query ,στην πράξη παρατηρούμε ότι η χρήση 1 worker είναι για ελάχιστα sec – γρηγορότερη.Αυτό μπορεί να οφείλεται στο γεγονός ότι απαιτείται περισσότερος χρόνος για το συντονισμό και την κατανομή του φόρτου εργασίας μεταξύ των workers αλλά και στη φύση των δεδομένων που έχουμε προς επεξεργασία(μέγεθος δεδομένων).