Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Η/Υ
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Βάσεων Γνώσεων και Δεδομένων
9ο Εξάμηνο
Ακαδημαϊκό Έτος 2022 - 2023

Προηγμένα Θέματα Βάσεων Δεδομένων

ΕΞΑΜΗΜΙΑΙΑ ΕΡΓΑΣΙΑ

Ανδρεάς Χρυσοβαλάντης- Κωνσταντίνος
03118102
el18102@mail.ntua.gr

Παπανικολάου Ιωάννης
03118064
el18064@mail.ntua.gr

Φεβρουάριος 2023

GitHub Link

https://github.com/ValantisAndreas/Advanced-DB

Εισαγωγή

Για την υλοποίηση της παρούσας εργασίας μας έχει παραχωρηθεί ένα cluster στον Okeano-Knossos, το οποίο αποτελείται από 2 υπολογιστές όπου ο καθένας διαθέτει 2 πυρήνες CPU, 4Gb ram και 30Gb μνήμη. Ακόμα έχουμε δημιουργήσει ένα υποδίκτυο με βάση τις οδηγίες που έχουν δοθεί (master + slave), στο οποίο η public IPv4 διεύθυνση για πρόσβαση από το διαδίκτυο είναι η **83.212.81.35**

Ακόμα χρησιμοποιήθηκαν και εγκαταστάθηκαν: python3.8, pip , PySpark, Apache Spark, Java και Apache Hadoop (για να έχουμε πρόσβαση στα hdfs αρχεία). Ακολουθήθηκαν οι οδηγίες που δόθηκαν και ότι υλικό βρέθηκε στον διαδικτύου.

Θα χρησιμοποιήσουμε το σύστημα HDFS για την αποθήκευση των input δεδομένων. Αυτό γίνεται μόνο μία φορά στην αρχή με τις παρακάτω εντολές:

1) Κατέβασμα τοπικά των αρχείων:

```
wget --no-check-certificate 'https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2022-01.parquet'
wget --no-check-certificate 'https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2022-02.parquet'
wget --no-check-certificate 'https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2022-03.parquet'
wget --no-check-certificate 'https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2022-04.parquet'
wget --no-check-certificate 'https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2022-05.parquet'
wget --no-check-certificate 'https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2022-06.parquet'
wget --no-check-certificate 'https://d37ci6vzurychx.cloudfront.net/misc/taxi+_zone_lookup.csv'
```

2) ‘Ανέβασμα’ στο Hdfs:

```
hadoop fs -put yellow_tripdata_2022-01.parquet hdfs://master:9000/files
hadoop fs -put yellow_tripdata_2022-02.parquet hdfs://master:9000/files
hadoop fs -put yellow_tripdata_2022-03.parquet hdfs://master:9000/files
hadoop fs -put yellow_tripdata_2022-04.parquet hdfs://master:9000/files
hadoop fs -put yellow_tripdata_2022-05.parquet hdfs://master:9000/files
hadoop fs -put yellow_tripdata_2022-06.parquet hdfs://master:9000/files
hadoop fs -put taxi+_zone_lookup.csv hdfs://master:9000/
```

Τέλος να αναφέρουμε πως για την απάντηση και την επίλυση των ερωτημάτων χρησιμοποιήθηκε η Apache **Spark** (και οι βιβλιοθήκες που περιέχει) και σαν κύρια γλώσσα η **Python**.

Ζητούμενο 1

Αφού εγκαταστάθηκε η πλατφόρμα εκτέλεσης Spark & HDFS προχωράμε στη δημιουργία δύο Dataframe και δύο RDD των parquet αρχείων taxi trips και του αρχείου taxi+_zone_lookup.csv. Επιπλέον, παρατηρήσαμε πώς υπήρχαν και εγγραφές εκτός του διαστήματος Ιανουαρίου-Ιουνίου τις οποίες διαγράψαμε. Τα δύο dataframe τα αποθηκεύσαμε στο hdfs ως αρχεία parquet (taxi_trips, zone_lookups) καθώς θα τα αξιοποιήσουμε στα επόμενα ερωτήματα. Η υλοποίηση των παραπάνω γίνεται στο αρχείο initial.py.

Ζητούμενο 2

A) Για το Q1

Φορτώνουμε τα αρχεία parquet σε δύο dataframe df1 (taxi_trips) και df2 (zone_lookups), από το hdfs. Στη συνέχεια φιλτράρουμε το df1, επιλέγοντας μόνο τις διαδρομές του Μαρτίου και αφετηρία το Battery Park (με βάση το df2 – location_id που βρίσκουμε). Τέλος, κατατάσσουμε και εμφανίζουμε την διαδρομή με το μεγαλύτερο φιλοδώρημα.

Οι χρόνοι αποτυπώνονται στον παρακάτω πίνακα:

Q1	Χρόνος (sec)
1 Worker	20,78
2 Workers	22,86

B) Για το Q2

Φορτώνουμε το parquet αρχείο taxi_trips στο df από το hdfs. Προσθέτουμε μια νέα στήλη με τον αριθμό του μήνα για κάθε διαδρομή και φιλτράρουμε τις διαδρομές με μηδενικό toll_amount. Επίσης, μέσω της συνάρτησης Window κάνουμε partitionBy('month'). Τέλος, για κάθε partition βρίσκουμε την διαδρομή με το υψηλότερο ποσό στα διόδια.

Οι χρόνοι αποτυπώνονται στον παρακάτω πίνακα:

Q2	Χρόνος (sec)
1 Worker	22,81
2 Workers	29,39

Ζητούμενο 3

A) Για το Q3 – Dataframe

Φορτώνουμε το parquet αρχείο taxi_trips στο df από το hdfs. Φιλτράρουμε τις διαδρομές όπου η αφετηρία είναι διαφορετική από το τέρμα και ομαδοποιούμε ανά δεκαπενθήμερο (με group-by και window). Να σημειωθεί πώς στη συνάρτηση αυτή έχουμε χρησιμοποιήσει ως παράμετρο τη startTime = “70 hours”, καθώς παρατηρήσαμε πώς η συνάρτηση ξεκινούσε να μετράει το πρώτο window 70 ώρες νωρίτερα από το timestamp 2022-01-01 00:00:00. Αυτό, οφείλεται στο γεγονός πώς η παράμετρος startTime είναι το offset με βάση την ημερομηνία 1970-01-01 00:00:00 UTC. Τέλος, εφαρμόζουμε τα κατάλληλα aggregations και ταξινομούμε σε αύξουσα σειρά τα windows που υπολογίσαμε.

Οι χρόνοι αποτυπώνονται στον παρακάτω πίνακα:

Q3 – Datafram/SQL	Χρόνος (sec)
1 Worker	24,01
2 Workers	30,54

Β) Για το Q3 – RDD API

Φορτώνουμε το parquet αρχείο taxi_trips ως ένα dataframe από το hdfs και το μετατρέπουμε σε rdd. Φιλτράρουμε τις τιμές με διαφορετικό σημείο αναχώρησης και άφιξης και μέσω της συνάρτησης map δημιουργούμε μια τούπλα (date, (total_amount, trip_distance)). Δημιουργούμε μια συνάρτηση fortnight ώστε να χωρίσουμε τις ημερομηνίες σε δεκαπενθήμερα και προσθέτουμε μια σταθερά 1 τούπλα που λειτουργεί ως counter των διαδρομών που έχουμε σε κάθε δεκαπενθήμερο. Τέλος, κάνουμε map την συνάρτηση στο rdd , υπολογίζουμε τον μέσο όρο για την απόσταση και το κόστος με reduceByKey και mapValues και με την sortByKey τα εμφανίζουμε σε αύξουσα σειρά.

Οι χρόνοι αποτυπώνονται στον παρακάτω πίνακα:

Q3 – RDD API	Χρόνος (sec)
1 Worker	218,28
2 Workers	184,34

Στα αποτελέσματα των δύο υλοποιήσεων με Dataframe και RDD API παρατηρούμε μικρές διαφορές οι οποίες λογικά οφείλονται στην αλλαγή της ώρας σε θερινή την οποία φαίνεται να λαμβάνει υπόψιν η συνάρτηση window.

Ζητούμενο 4

Α) Για το Q4

Φορτώνουμε το parquet αρχείο taxi_trips ως ένα dataframe από το hdfs. Στην συνέχεια προσθέτομε τις στήλες “day_of_week” και “hour” τις οποίες κάνουμε select μαζί με την στήλη “Passanger_count”. Μέσω της συνάρτησης Window.partionBy χωρίζουμε τα δεδομένα ανά ημέρα και με βάση τη στήλη “Passanger_count” βρίσκουμε τις 3 ώρες αιχμής ανά ημέρα.

Οι χρόνοι αποτυπώνονται στον παρακάτω πίνακα:

Q4	Χρόνος (sec)
1 Worker	33,65
2 Workers	45,12

Α) Για το Q5

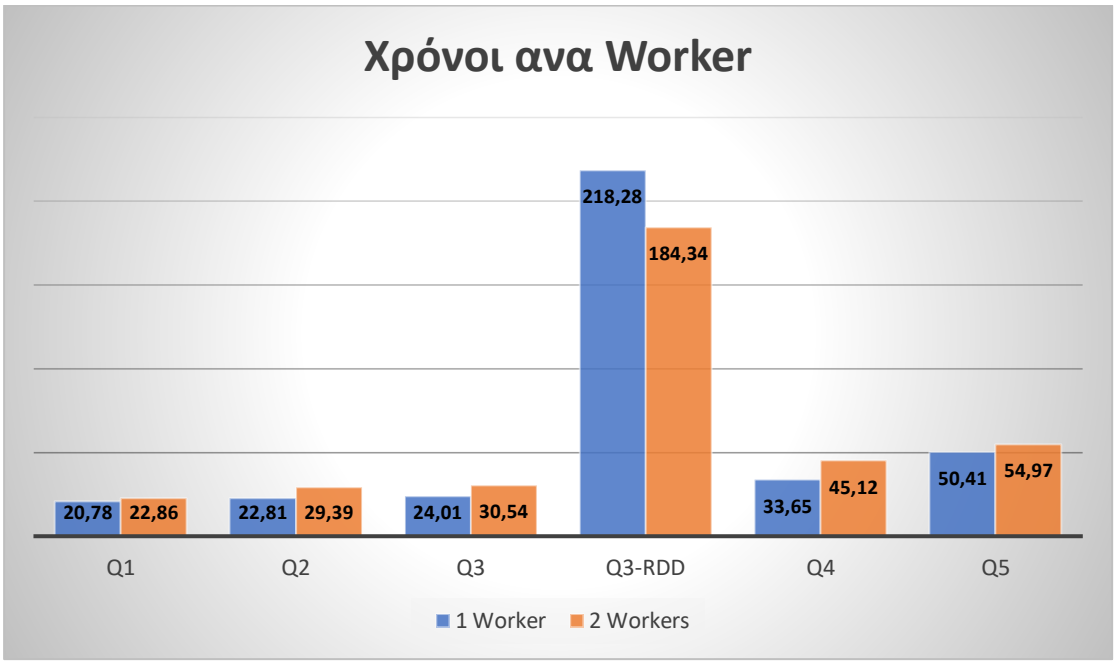
Φορτώνουμε το parquet αρχείο taxi_trips ως ένα dataframe από το hdfs. ‘Επειτα, προσθέτομε τις στήλες “day”, “percent_tip” και “month” τις οποίες κάνουμε select. Για τις διαδρομές που το fare_amount είναι μηδενικό θεωρήσαμε πώς και το percent_tip είναι μηδενικό. Στη συνέχεια, αξιοποιώντας την συνάρτηση Window.partitionBy βρίσκουμε και κρατάμε μόνο το μεγαλύτερο φιλοδώρημα κάθε ημερομηνίας. Τέλος, βρίσκουμε τις 5 κορυφαίες μέρες κάθε μήνα.

Οι χρόνοι αποτυπώνονται στον παρακάτω πίνακα:

Q5	Χρόνος (sec)
1 Worker	50,41
2 Workers	54.97

Συνολικά οι χρόνοι για όλα τα ερωτήματα συγκεντρώνονται στον παρακάτω πίνακα.

Time (sec)	**With 1 worker**	**With 2 workers**
Q1	*20,78*	22,86
Q2	*22,81*	29,39
Q3 – Dataframe	*24,01*	30,54
Q3 – RDD API	*218,28*	184,34
Q4	*33,65*	45,12
Q5	*50,41*	54,97

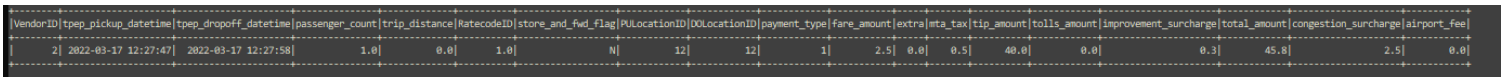


Παρατηρήσεις για τους χρόνους

- Ο χρόνος εκτέλεσης Q3-RDD είναι πολλαπλάσιος σε σχέση με τον Q3-Dataframe. Σε αντίθεση με τα υπόλοιπα ο χρόνος με 2 workers είναι μικρότερος σε σχέση με 1 worker.
- Ο χρόνος εκτέλεσης για 2 workers είναι μεγαλύτερος σε σχέση με 1 worker. Θα αναμέναμε να συμβαίνει το αντίθετο. Ωστόσο, αυτό ενδεχομένως να οφείλεται σε θέματα δικτύου.

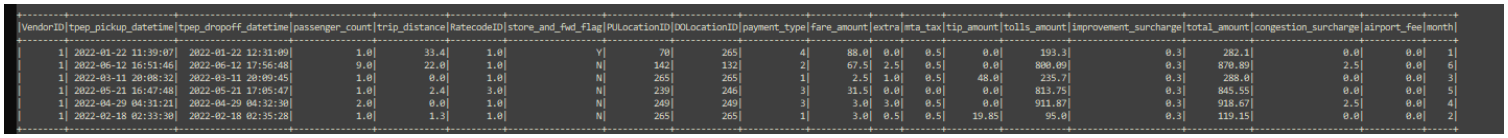
Αποτελέσματα Ερωτημάτων

Q1



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
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

Q2



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
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	1
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	6
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	3
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	5
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	4
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	2

Q3

window	Average_Trip_Distance	Average_Cost
{2022-01-01 00:00:00, 2022-01-16 00:00:00	5.58	19.9
{2022-01-16 00:00:00, 2022-01-31 00:00:00}	4.8	19.04
{2022-01-31 00:00:00, 2022-02-15 00:00:00	5.95	19.55
{2022-02-15 00:00:00, 2022-03-02 00:00:00,	6.19	20.17
{2022-03-02 00:00:00, 2022-03-17 00:00:00}	6.61	20.69
{2022-03-17 00:00:00, 2022-04-01 01:00:00}	5.52	21.12
{2022-04-01 01:00:00, 2022-04-16 01:00:00}	5.68	21.51
{2022-04-16 01:00:00, 2022-05-01 01:00:00}	5.8	21.43
{2022-05-01 01:00:00, 2022-05-16 01:00:00}	6.26	21.93
{2022-05-16 01:00:00, 2022-05-31 01:00:00}	8.0	22.81
{2022-05-31 01:00:00, 2022-06-15 01:00:00}	6.37	22.44
{2022-06-15 01:00:00, 2022-06-30 01:00:00}	6.15	22.35
{2022-06-30 01:00:00, 2022-06-30 23:59:59}	5.95	22.24

Q3 -RDD

15 Day Period Number:	1	Average Trip Cost:	19.90	Average Trip Distance:	5.58
15 Day Period Number:	2	Average Trip Cost:	19.04	Average Trip Distance:	4.80
15 Day Period Number:	3	Average Trip Cost:	19.55	Average Trip Distance:	5.95
15 Day Period Number:	4	Average Trip Cost:	20.17	Average Trip Distance:	6.19
15 Day Period Number:	5	Average Trip Cost:	20.69	Average Trip Distance:	6.61
15 Day Period Number:	6	Average Trip Cost:	21.08	Average Trip Distance:	5.50
15 Day Period Number:	7	Average Trip Cost:	21.52	Average Trip Distance:	5.68
15 Day Period Number:	8	Average Trip Cost:	21.43	Average Trip Distance:	5.80
15 Day Period Number:	9	Average Trip Cost:	21.92	Average Trip Distance:	6.25
15 Day Period Number:	10	Average Trip Cost:	22.81	Average Trip Distance:	8.00
15 Day Period Number:	11	Average Trip Cost:	22.45	Average Trip Distance:	6.38
15 Day Period Number:	12	Average Trip Cost:	22.35	Average Trip Distance:	6.15
15 Day Period Number:	13	Average Trip Cost:	21.91	Average Trip Distance:	5.93

Q4

day_of_week	hour	Passenger_count
Wednesday	9	9.0
Wednesday	12	8.0
Wednesday	20	8.0
Tuesday	21	9.0
Tuesday	22	9.0
Tuesday	20	9.0
Friday	19	9.0
Friday	18	9.0
Friday	8	8.0
Thursday	2	9.0
Thursday	21	9.0
Thursday	0	8.0
Saturday	6	9.0
Saturday	14	9.0
Saturday	13	9.0
Monday	20	9.0
Monday	1	9.0
Monday	20	9.0
Sunday	16	9.0
Sunday	15	8.0
Sunday	17	8.0

Q5

day	month	percent_tip
03/01/2022	1	250000.0
31/01/2022	1	1100000.0
16/01/2022	1	120000.0
01/01/2022	1	500000.0
09/01/2022	1	1688800.0
27/02/2022	2	250000.0
21/02/2022	2	450000.0
13/02/2022	2	296900.0
09/02/2022	2	250000.0
24/02/2022	2	150000.0
18/03/2022	3	1000000.0
21/03/2022	3	700000.0
26/03/2022	3	200000.0
05/03/2022	3	200000.0
19/03/2022	3	100000.0
03/04/2022	4	400000.0
02/04/2022	4	1250000.0
30/04/2022	4	200000.0
12/04/2022	4	2250000.0
21/04/2022	4	1200000.0
15/05/2022	5	200000.0
16/05/2022	5	200000.0
06/05/2022	5	120000.0
12/05/2022	5	750000.0
20/05/2022	5	650000.0
25/06/2022	6	1550000.0
13/06/2022	6	1500000.0
16/06/2022	6	570000.0
10/06/2022	6	650000.0
18/06/2022	6	400000.0