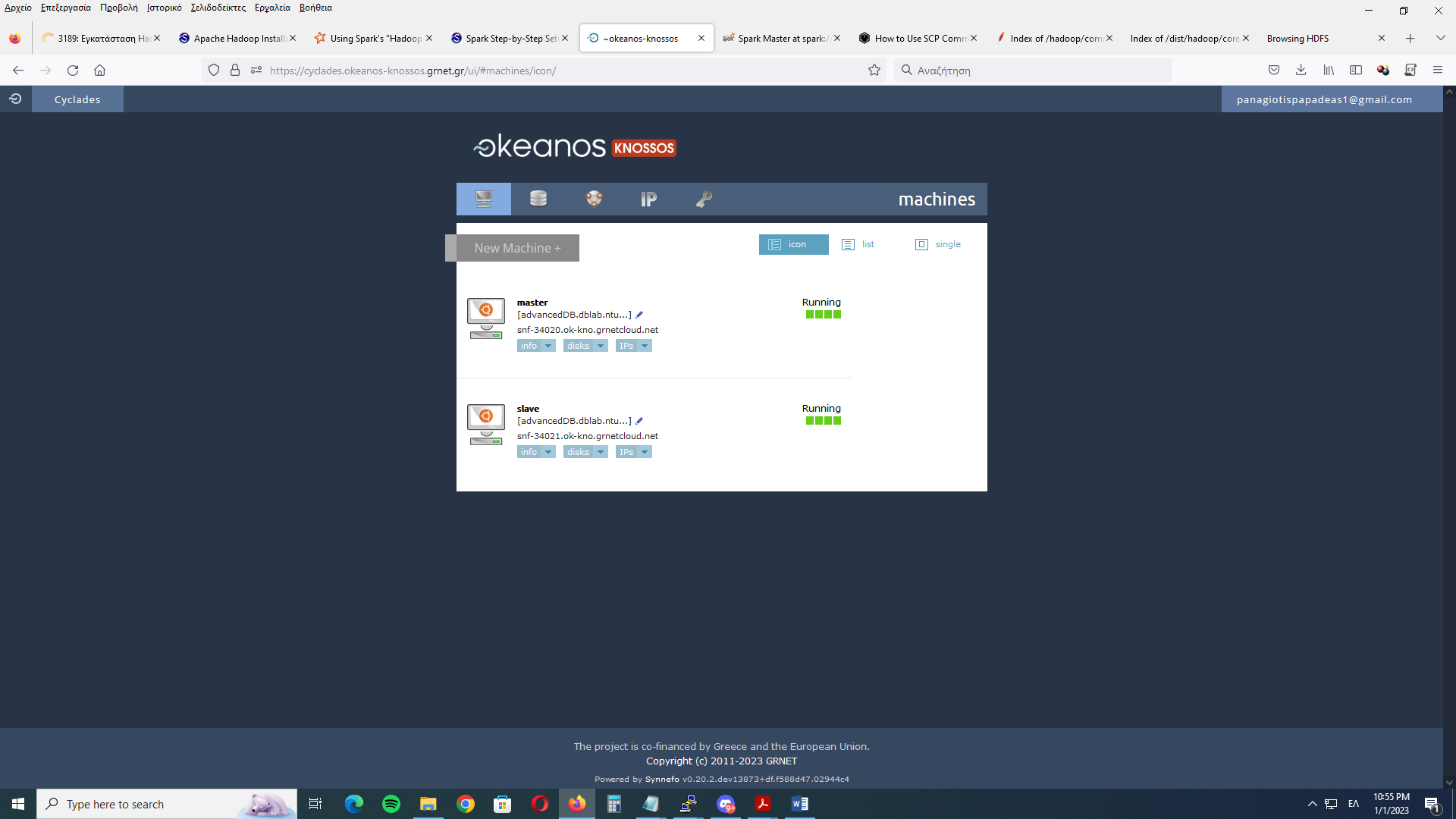
Προχωρημένα θέματα βάσεων δεδομένων

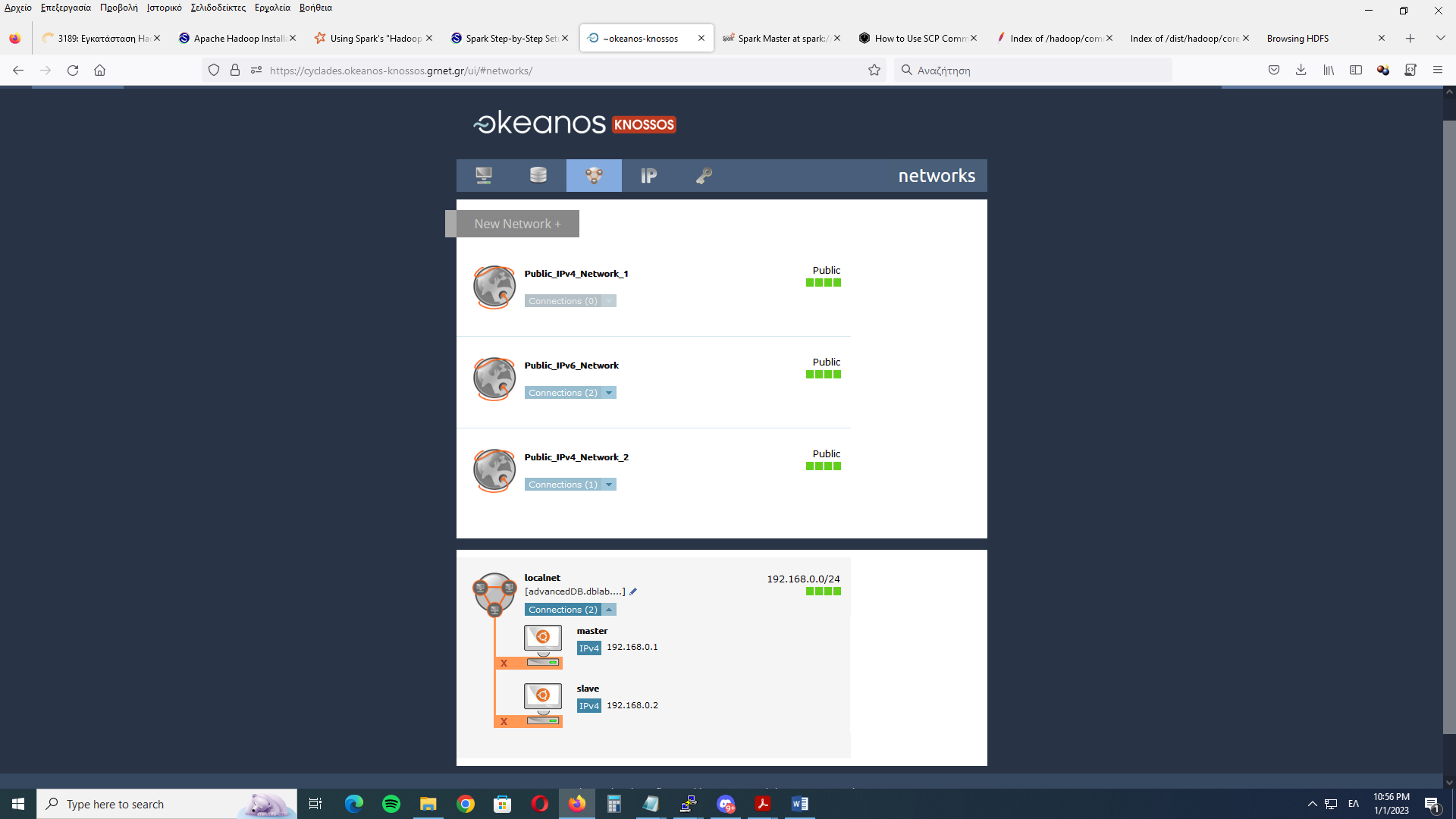
Εξαμηνιαία εργασία

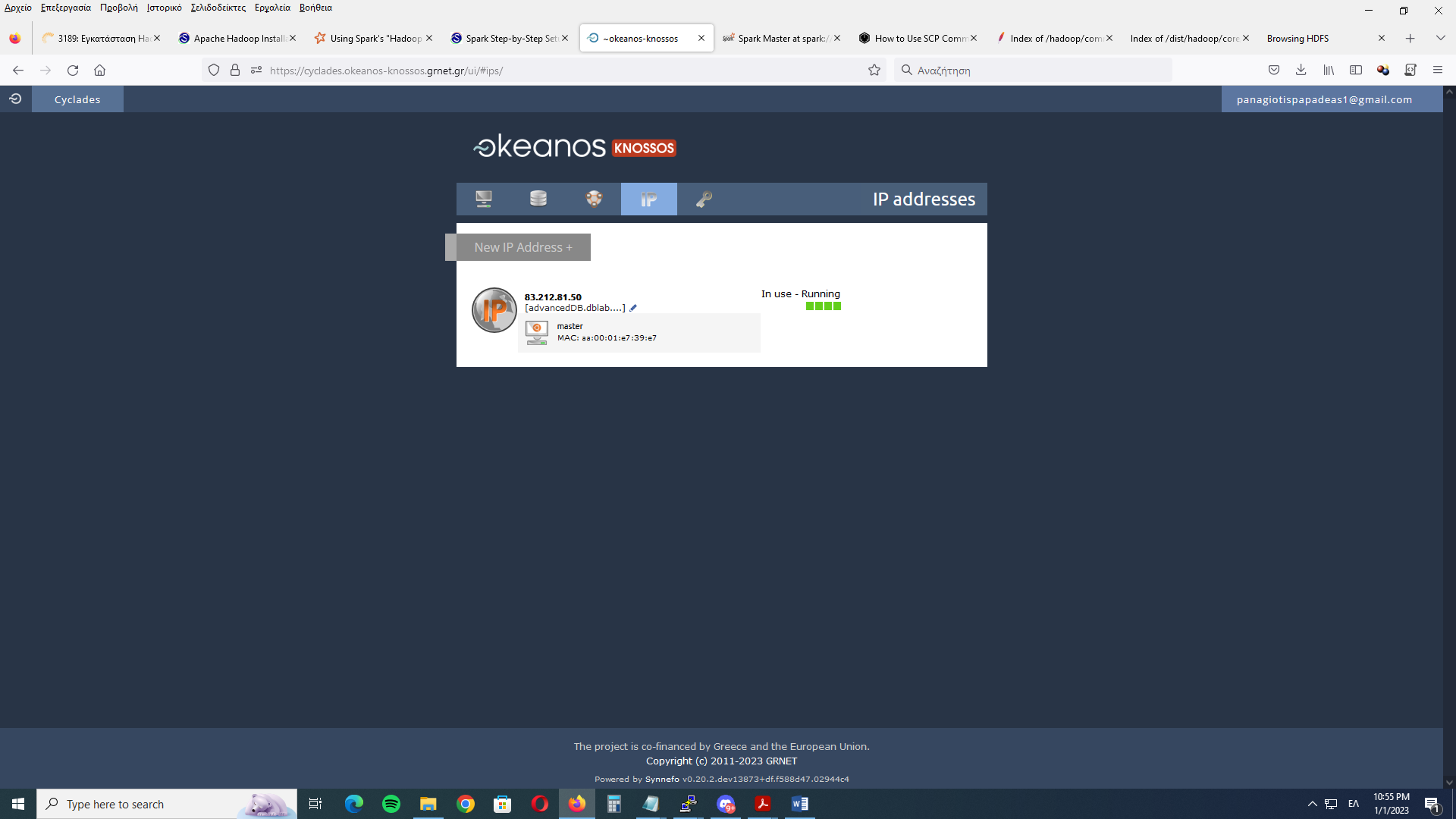
ΠΑΝΑΓΙΩΤΗΣ ΠΑΠΑΔΕΑΣ Α.Μ. 03118039

ΚΡΙΣ ΚΟΥΤΣΗ Α.Μ. 03118905

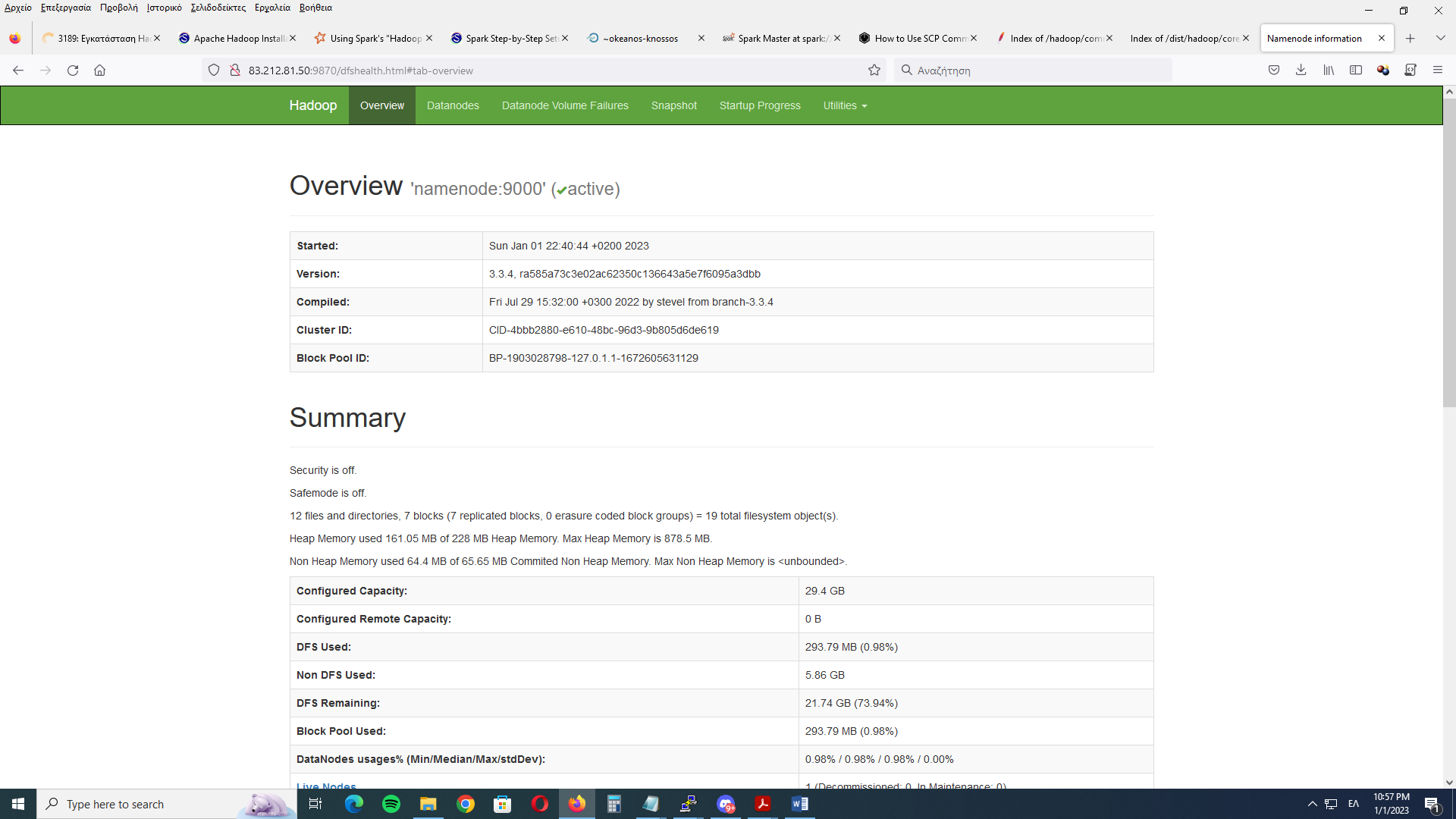
Αρχικά κατασκευάζουμε τα vms στο okeanos σύμφωνα με τις οδηγίες του pdf. Δημιουργούμε ένα master node και ένα slave, τα οργανώνουμε σε τοπικό δίκτυο για να μπορούν να επικοινωνούν μεταξύ τους και αναθέτουμε μια public IP στο master node.

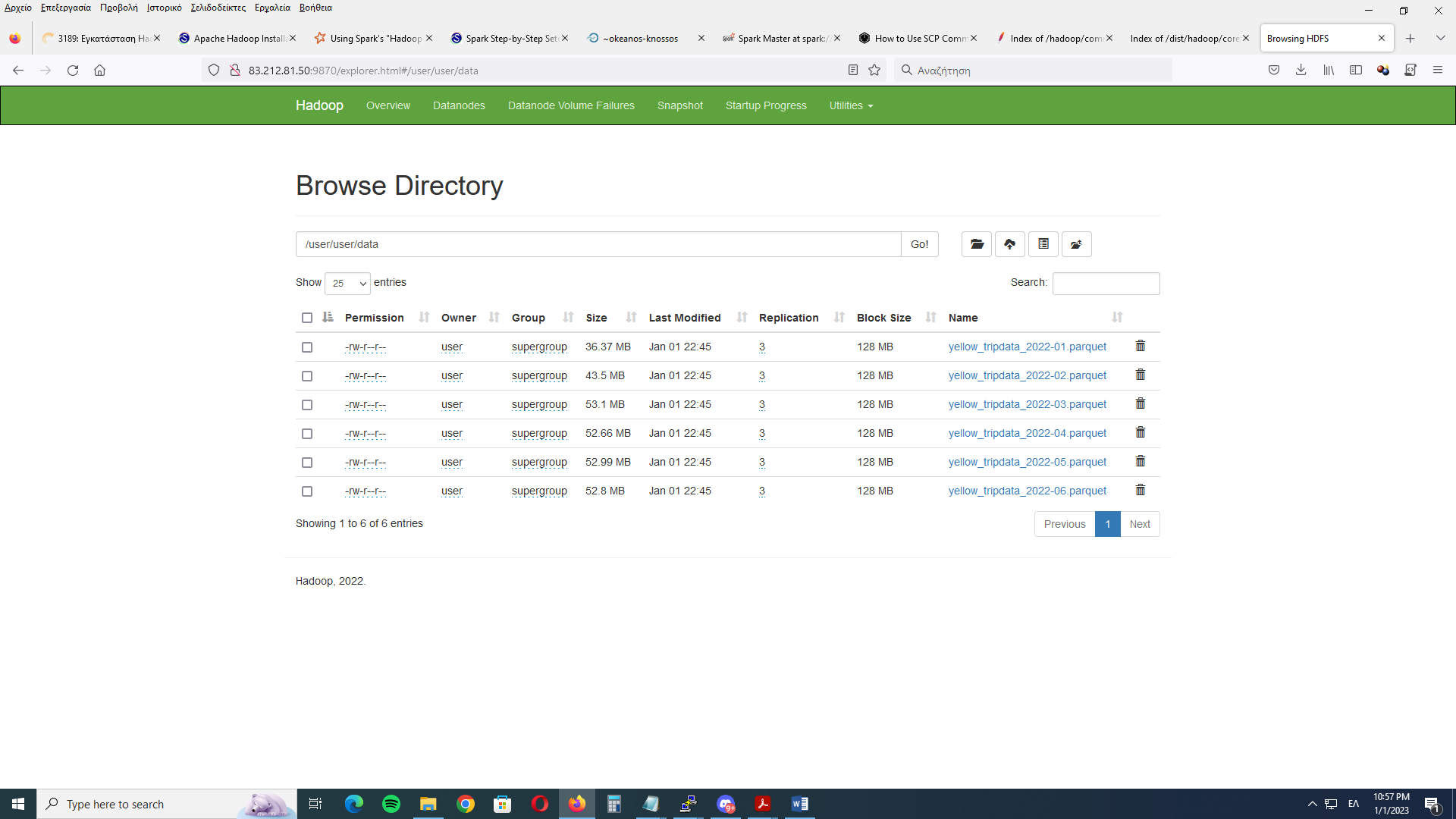


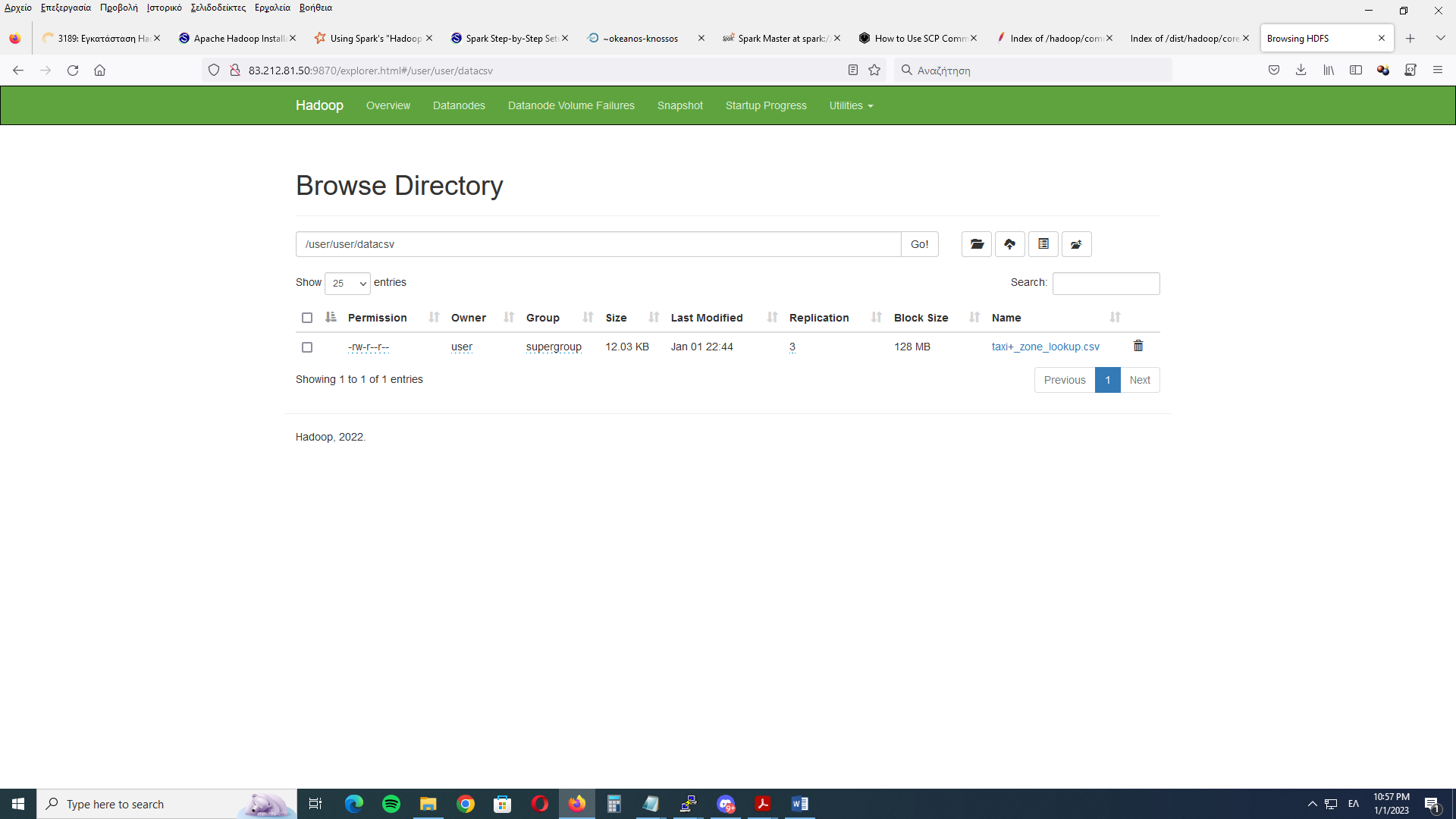




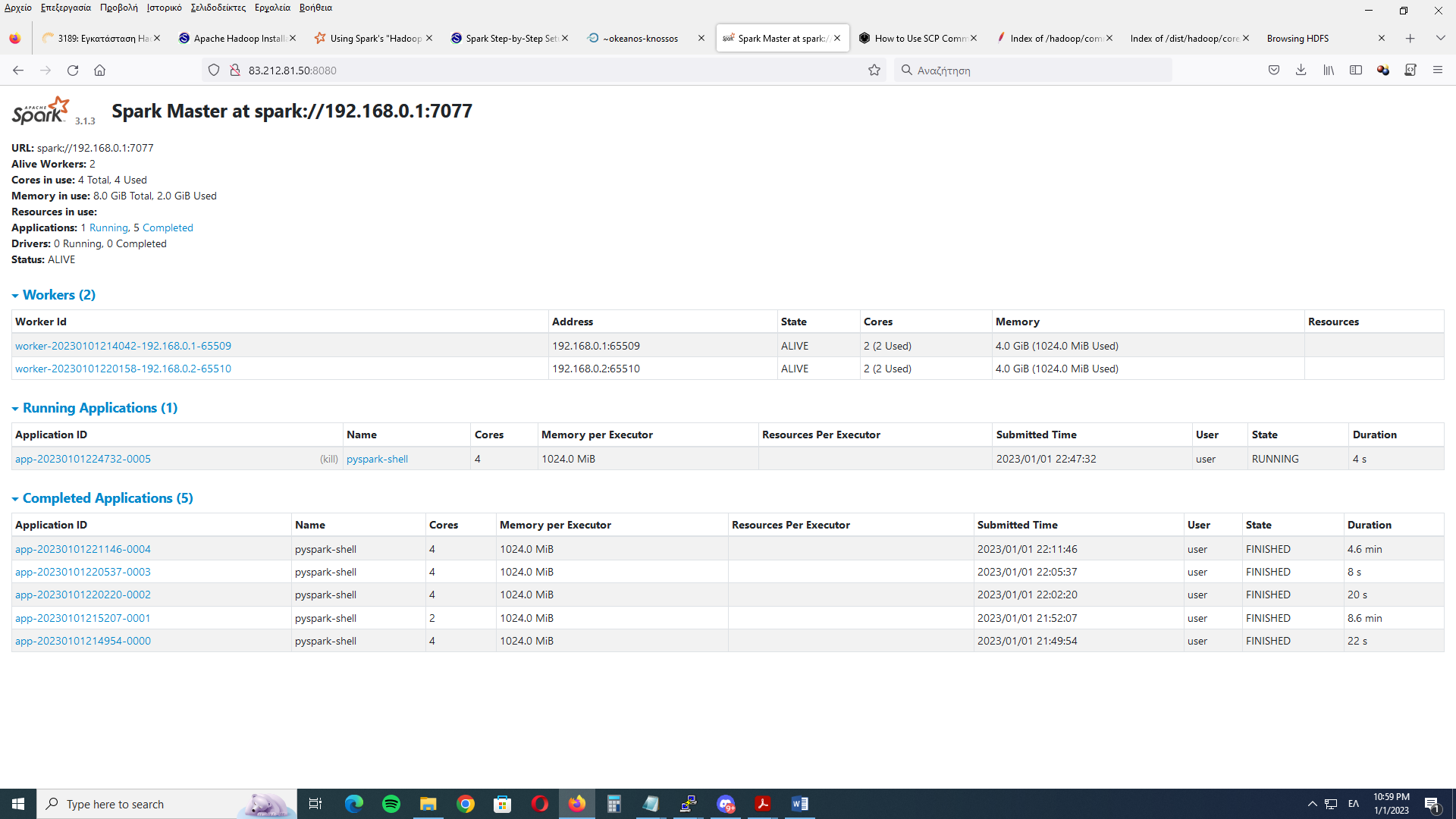
Στη συνέχεια εγκαθιστούμε το Hadoop (HDFS). Ορίζουμε ως namenode το master και ως datanodes το master και το slave. Ανεβάζουμε τα αρχεία με τα δεδομένα μέσω της εντολής: hdfs dfs –put στα αντίστοιχα folders και χτυπάμε την public διεύθυνση του Master (83.212.81.50) στη θύρα 9870 για να δούμε ότι το hdfs έχει στηθεί σωστά και υπάρχουν τα αντίστοιχα αρχεία



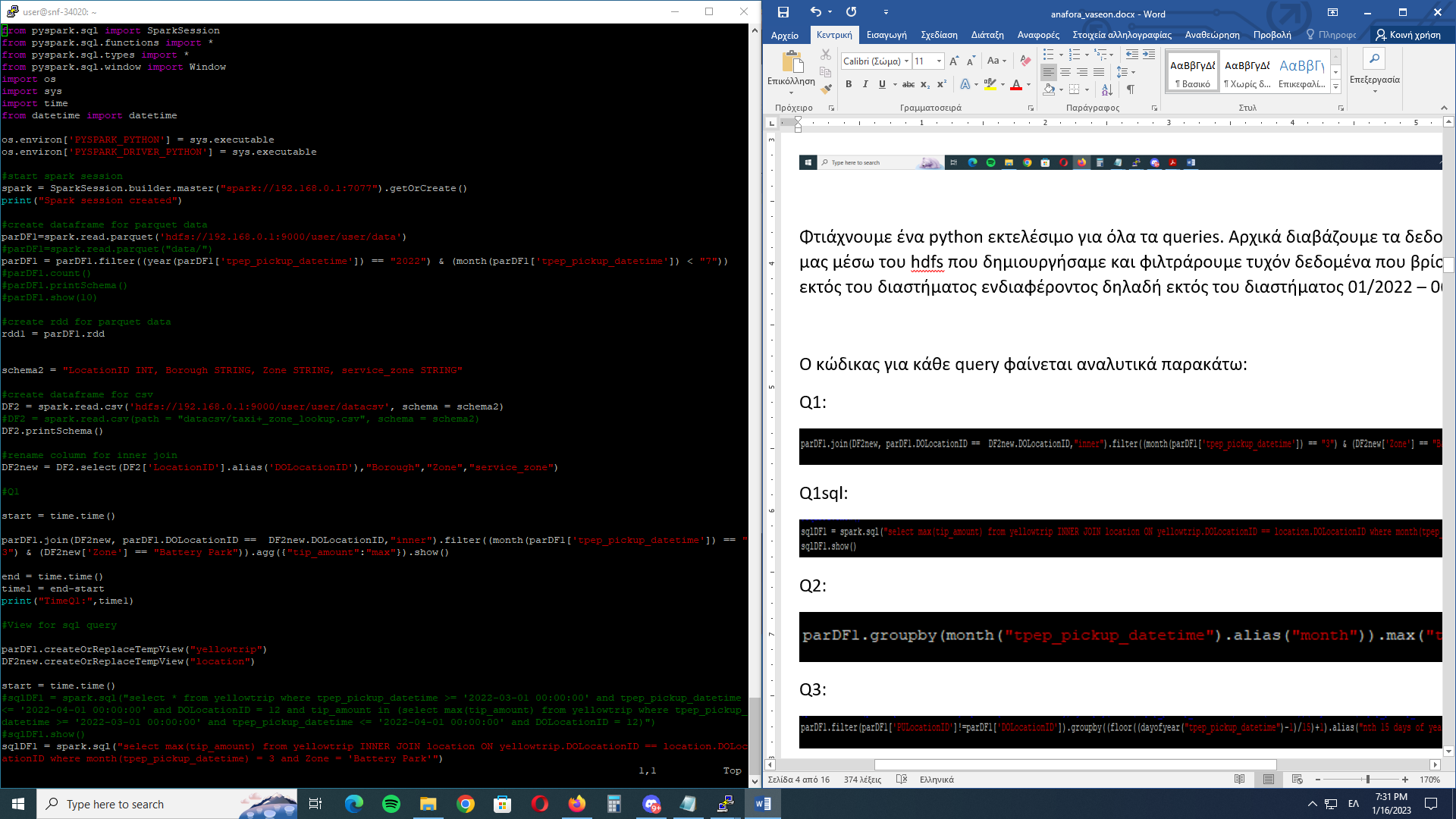




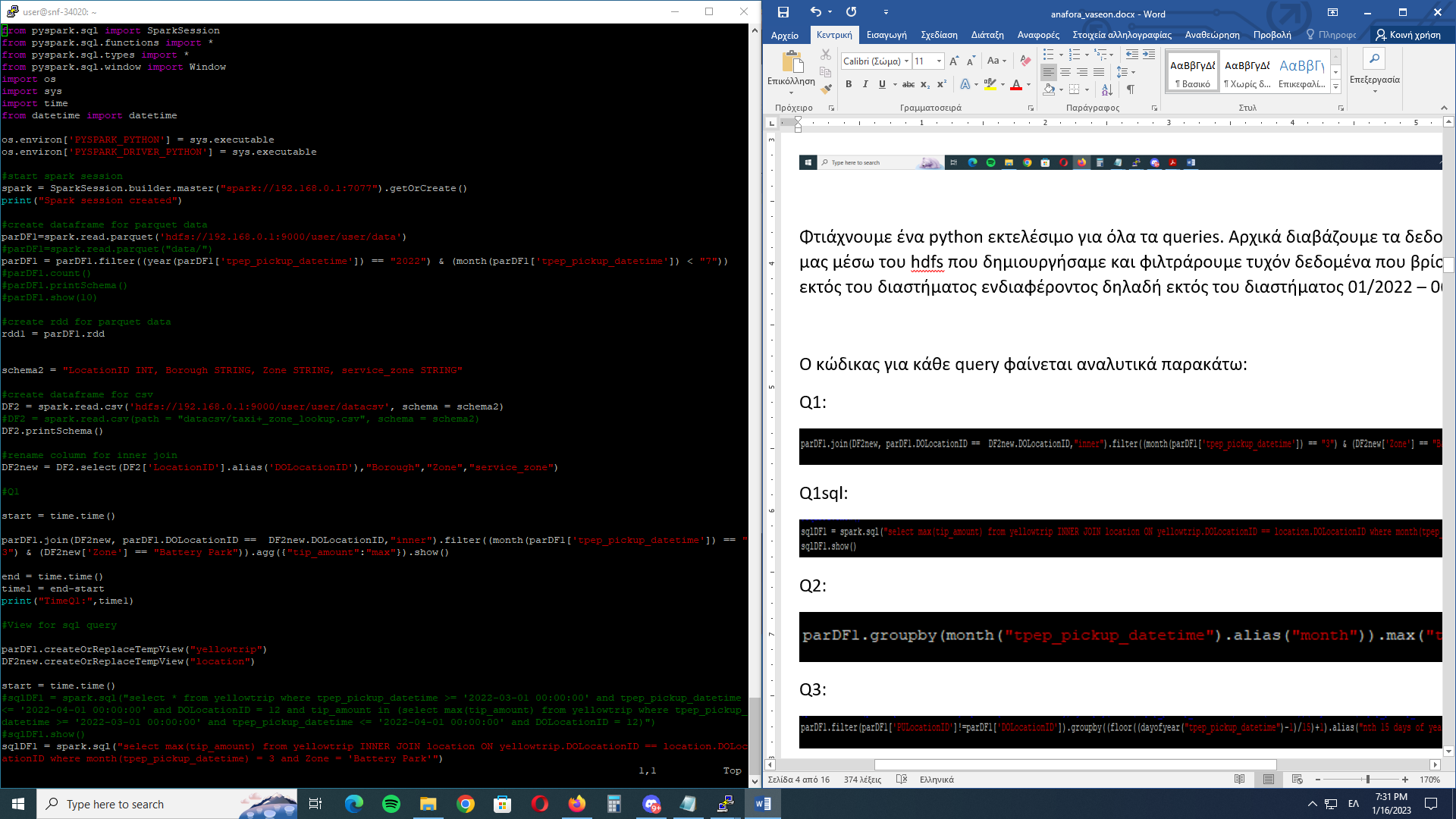
Αντίστοιχα εγκαθιστούμε το apache spark. Ορίζουμε ως master node το master και ως workers τον master και τον slave. Αργότερα απενεργοποιούμε τον ένα worker για να δούμε τη διαφορά του χρόνου στα queries. Χτυπάμε την public διεύθυνση του Master (83.212.81.50) στη θύρα 8080 και βλέπουμε ότι το apache spark έχει στηθεί σωστά και υπάρχουν 2 ενεργοί workers.



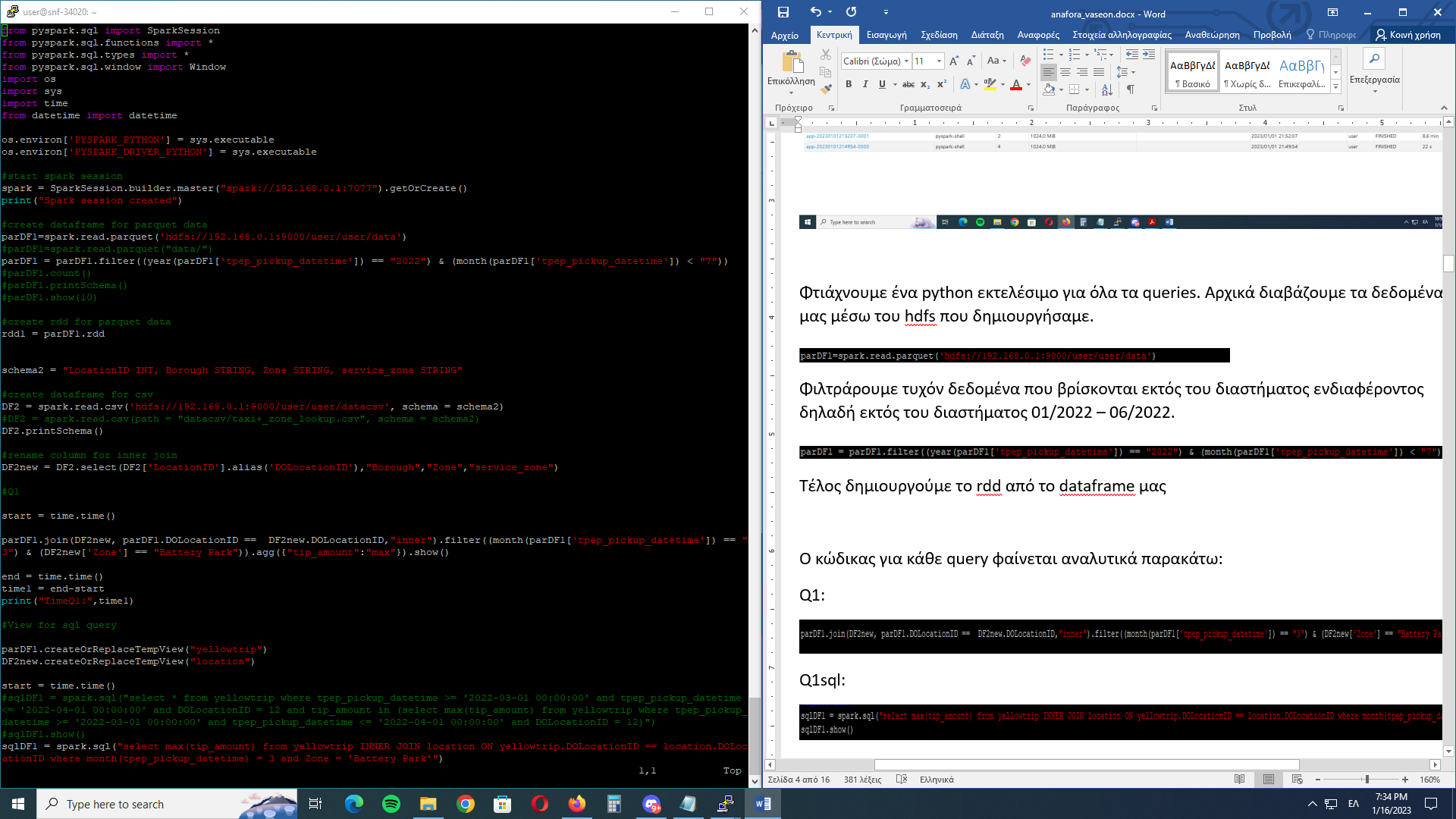
Φτιάχνουμε ένα python εκτελέσιμο για όλα τα queries. Αρχικά διαβάζουμε τα δεδομένα μας μέσω του hdfs που δημιουργήσαμε.



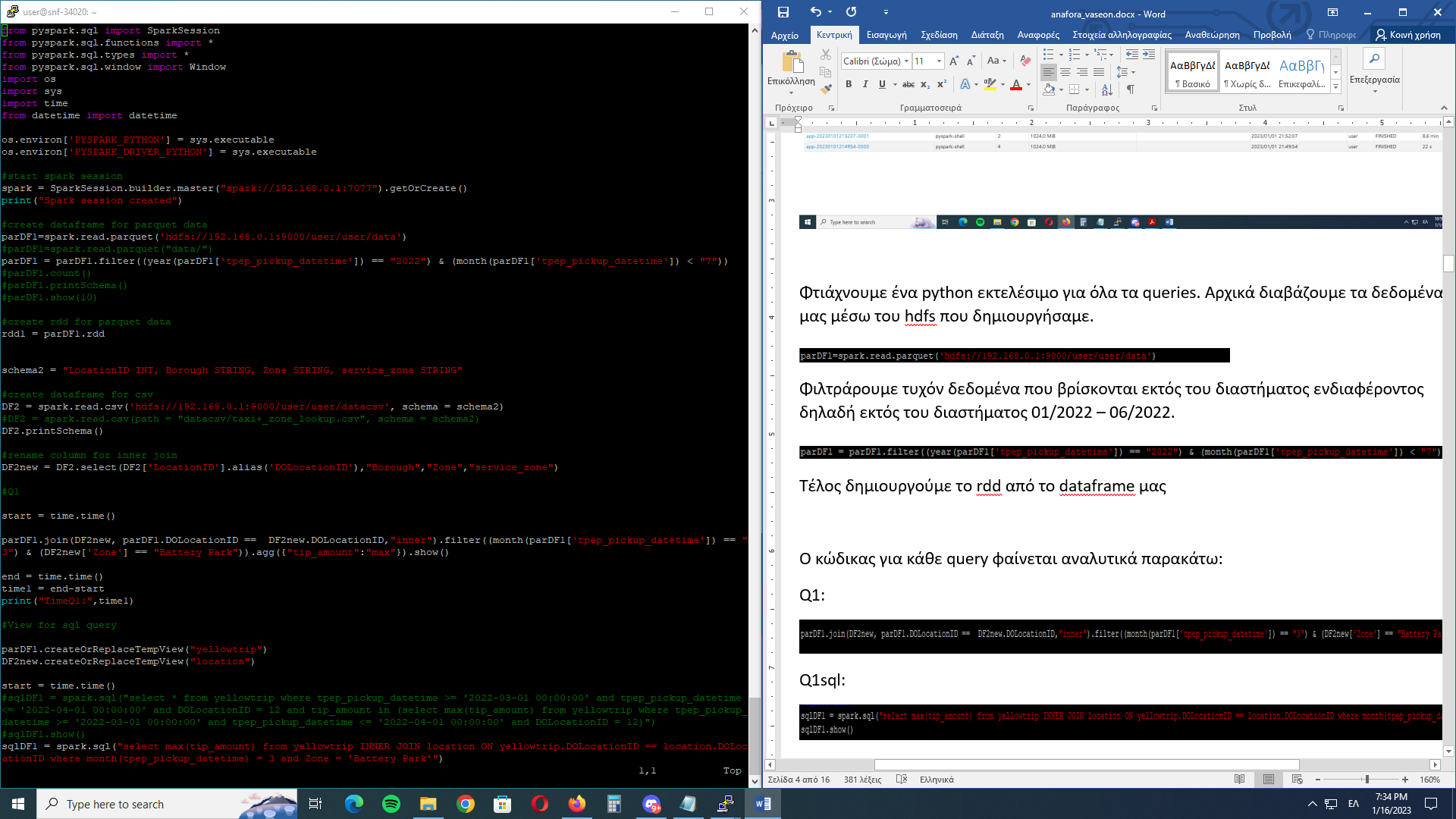
Φιλτράρουμε τυχόν δεδομένα που βρίσκονται εκτός του διαστήματος ενδιαφέροντος δηλαδή εκτός του διαστήματος 01/2022 – 06/2022.



Τέλος δημιουργούμε το rdd από το dataframe μας.

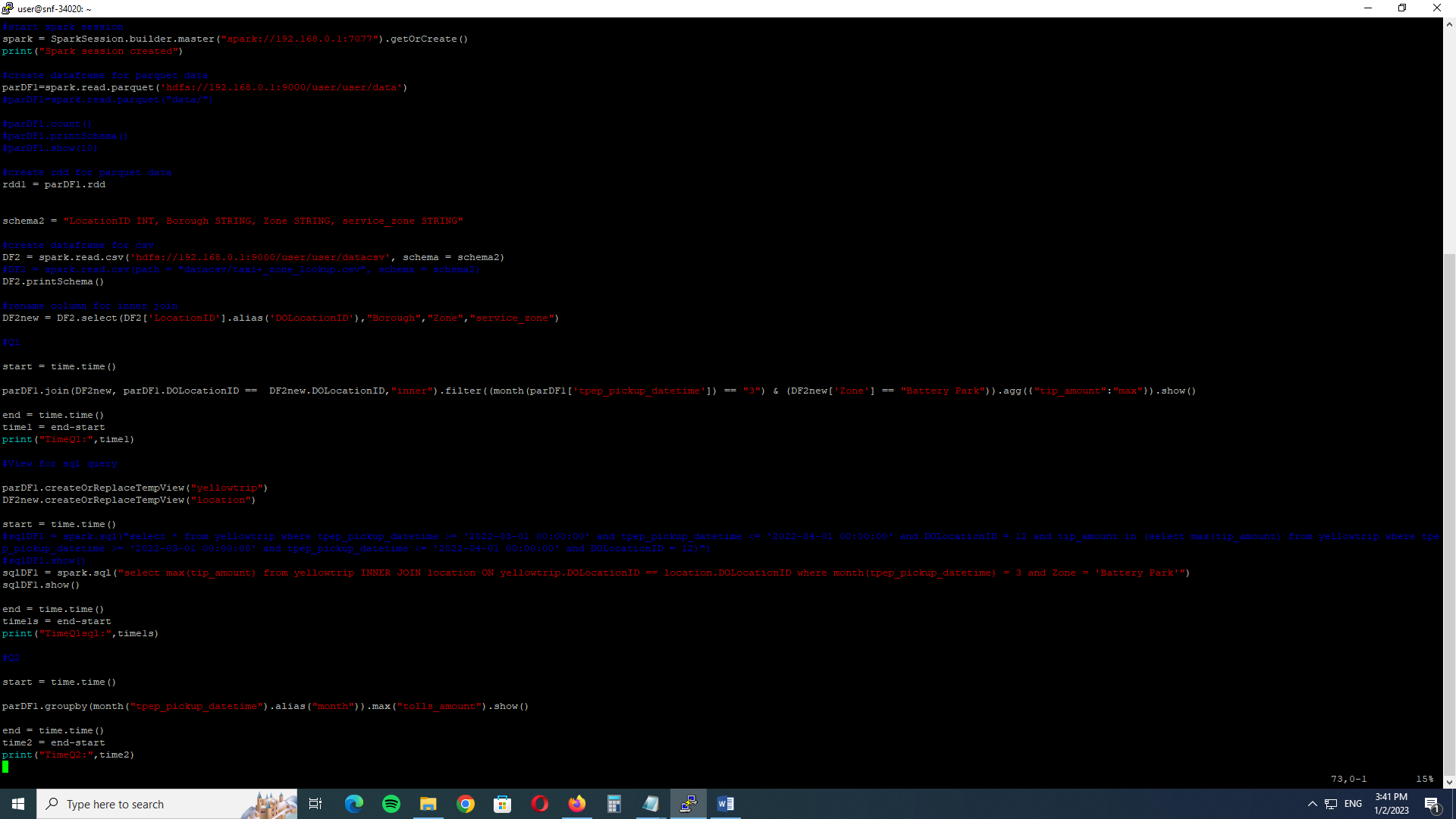


Αντίστοιχα για το csv.

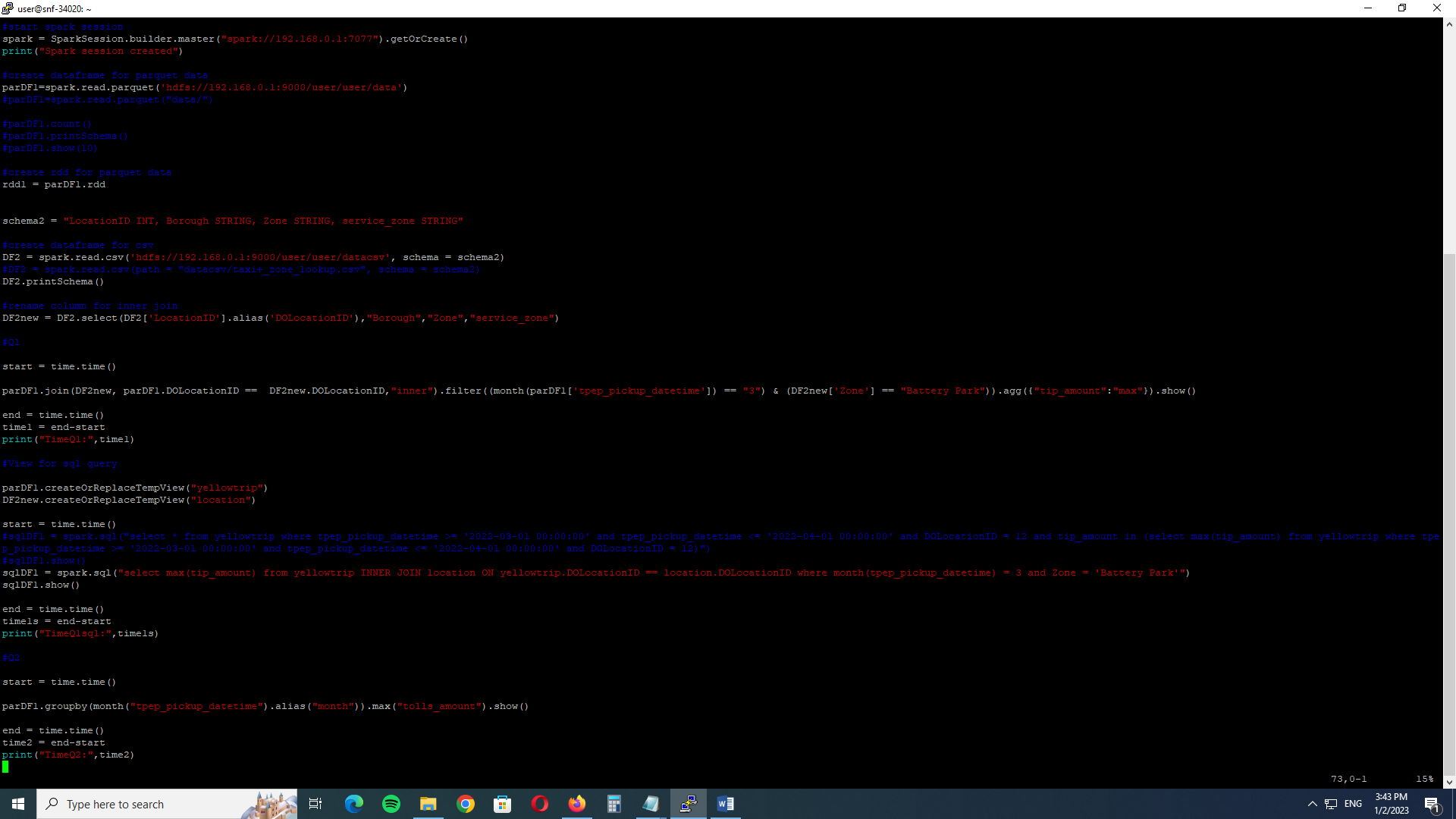


Ο κώδικας για κάθε query φαίνεται αναλυτικά παρακάτω:

Q1:



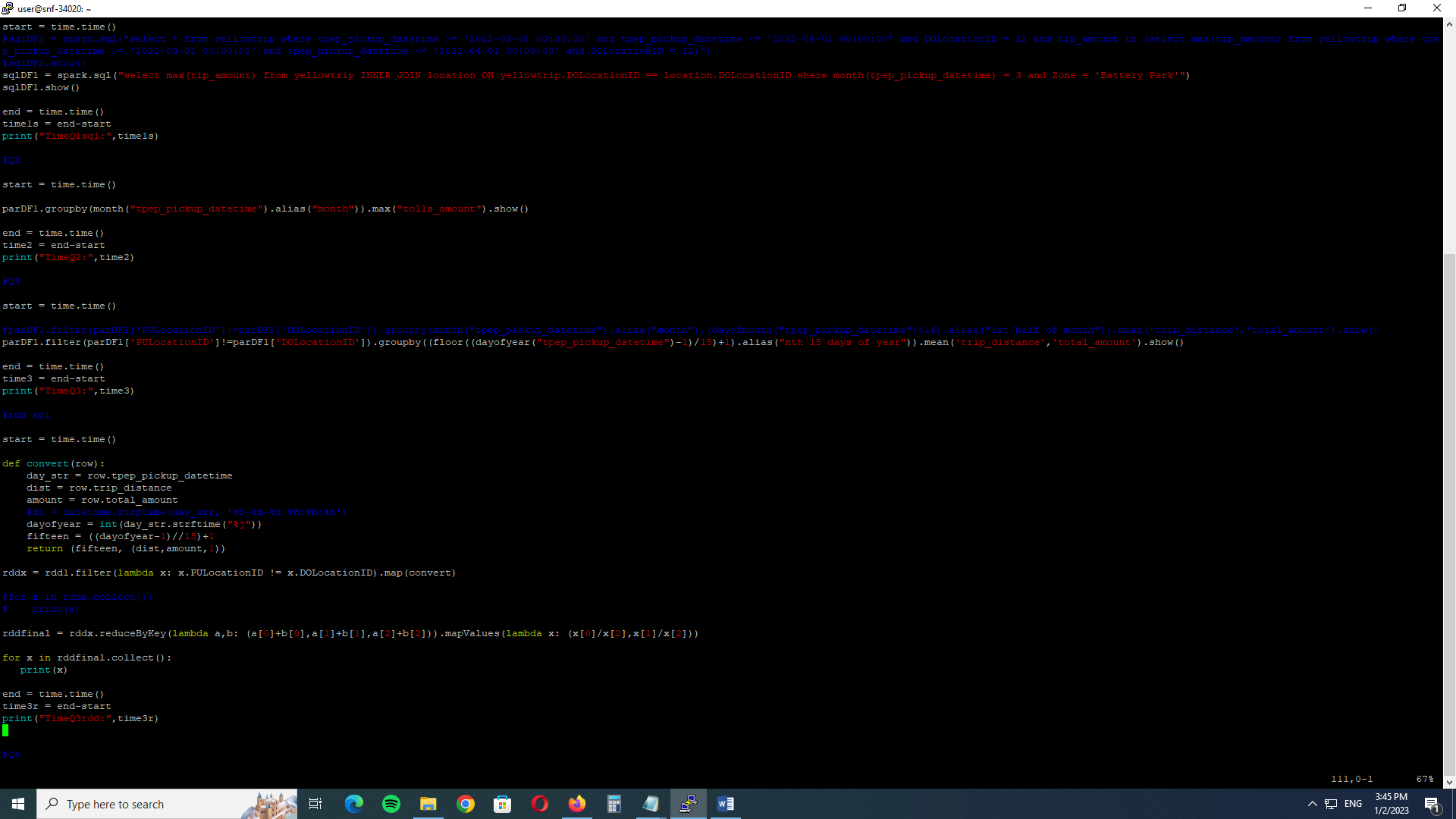
Q1sql:



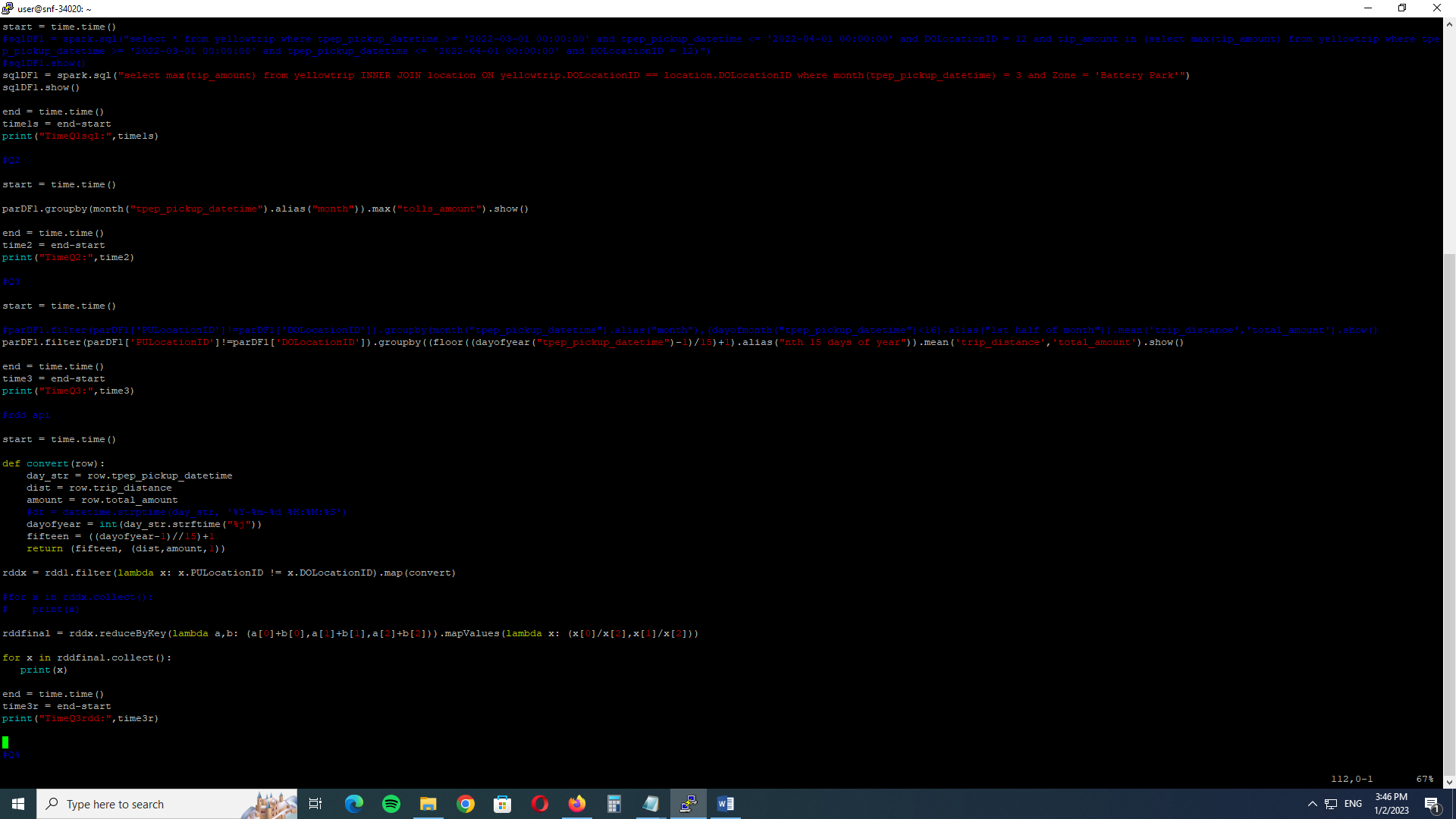
Q2:



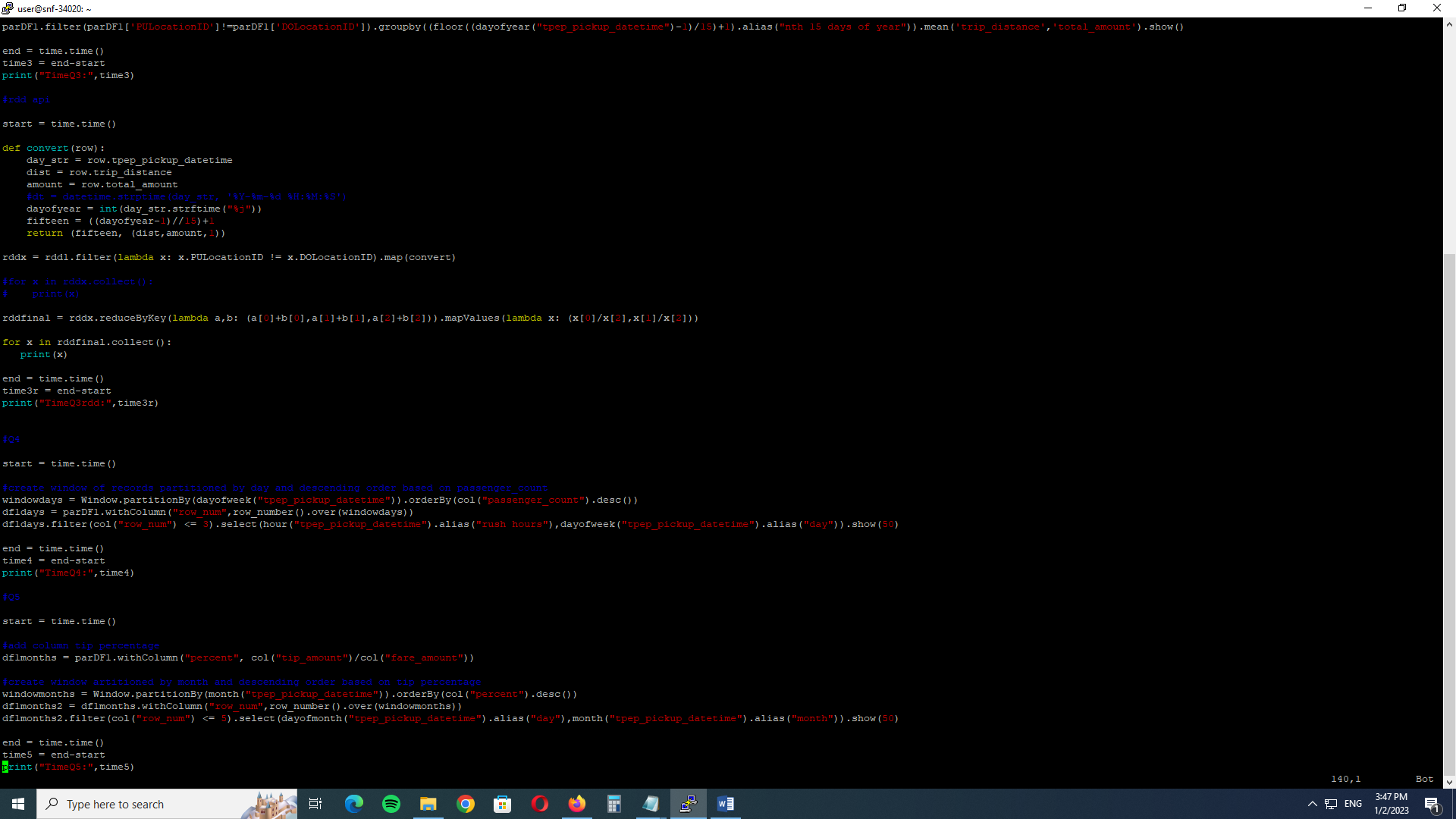
Q3:



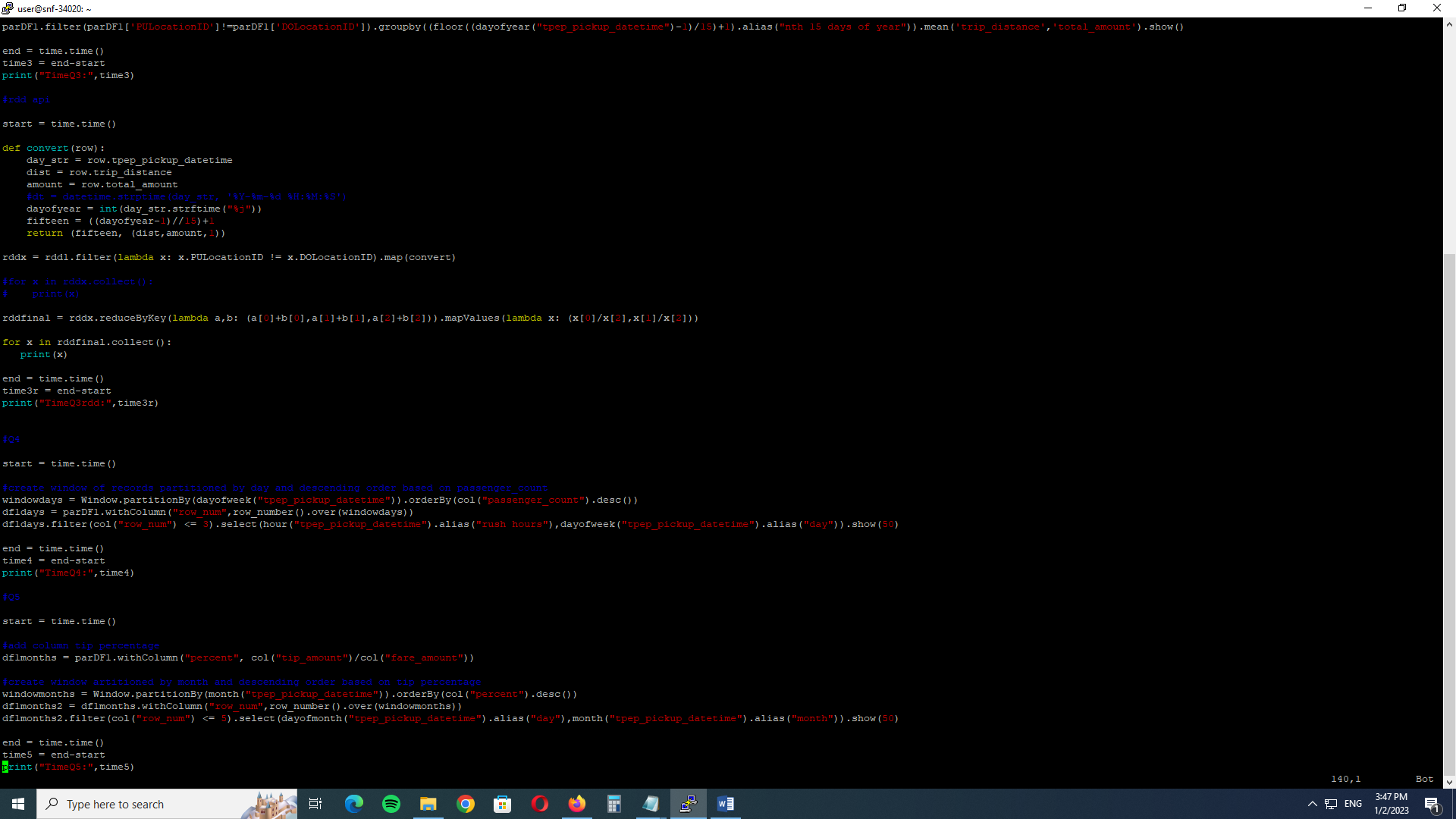
Q3rdd:



Q4:

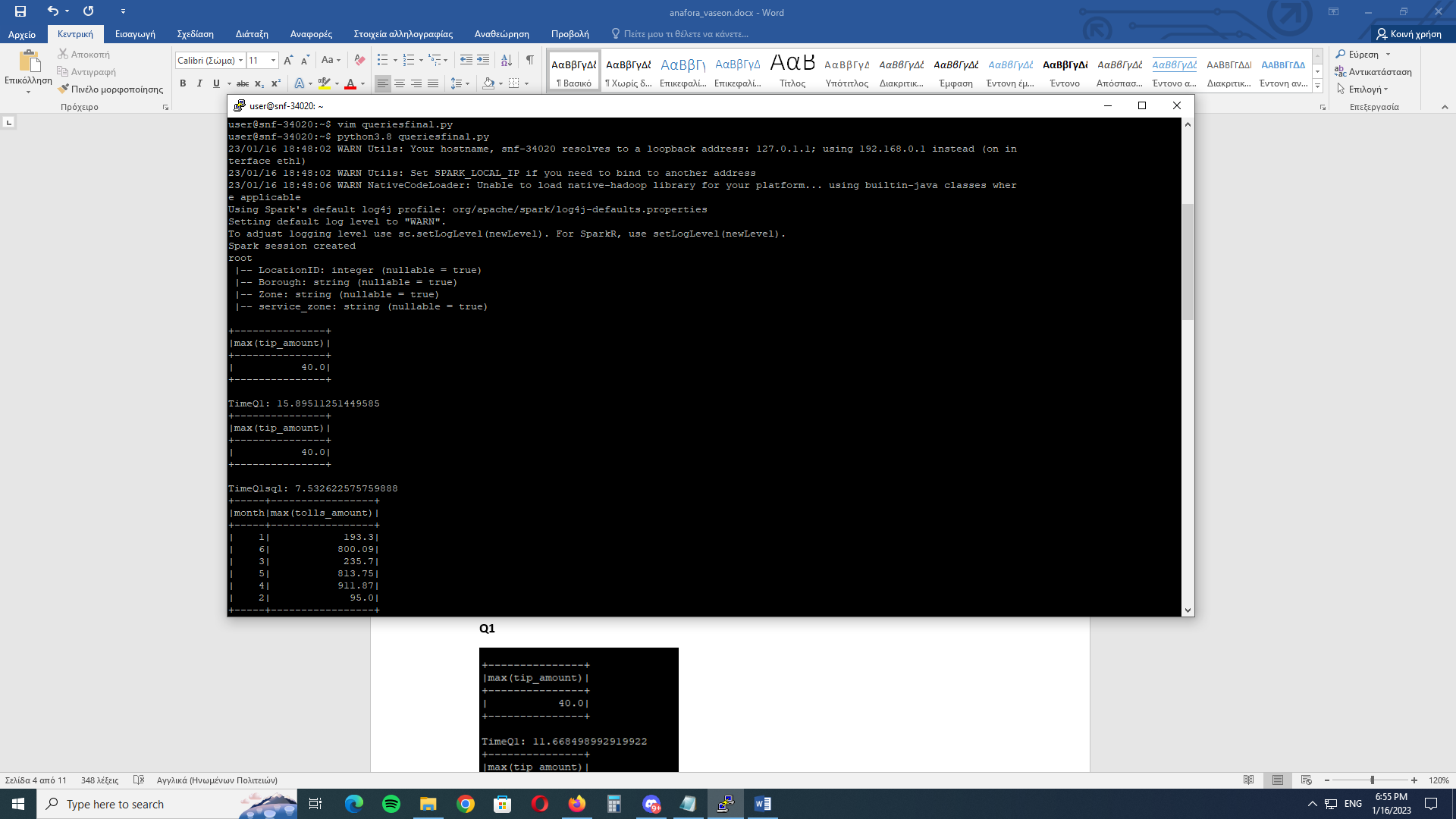


Q5:

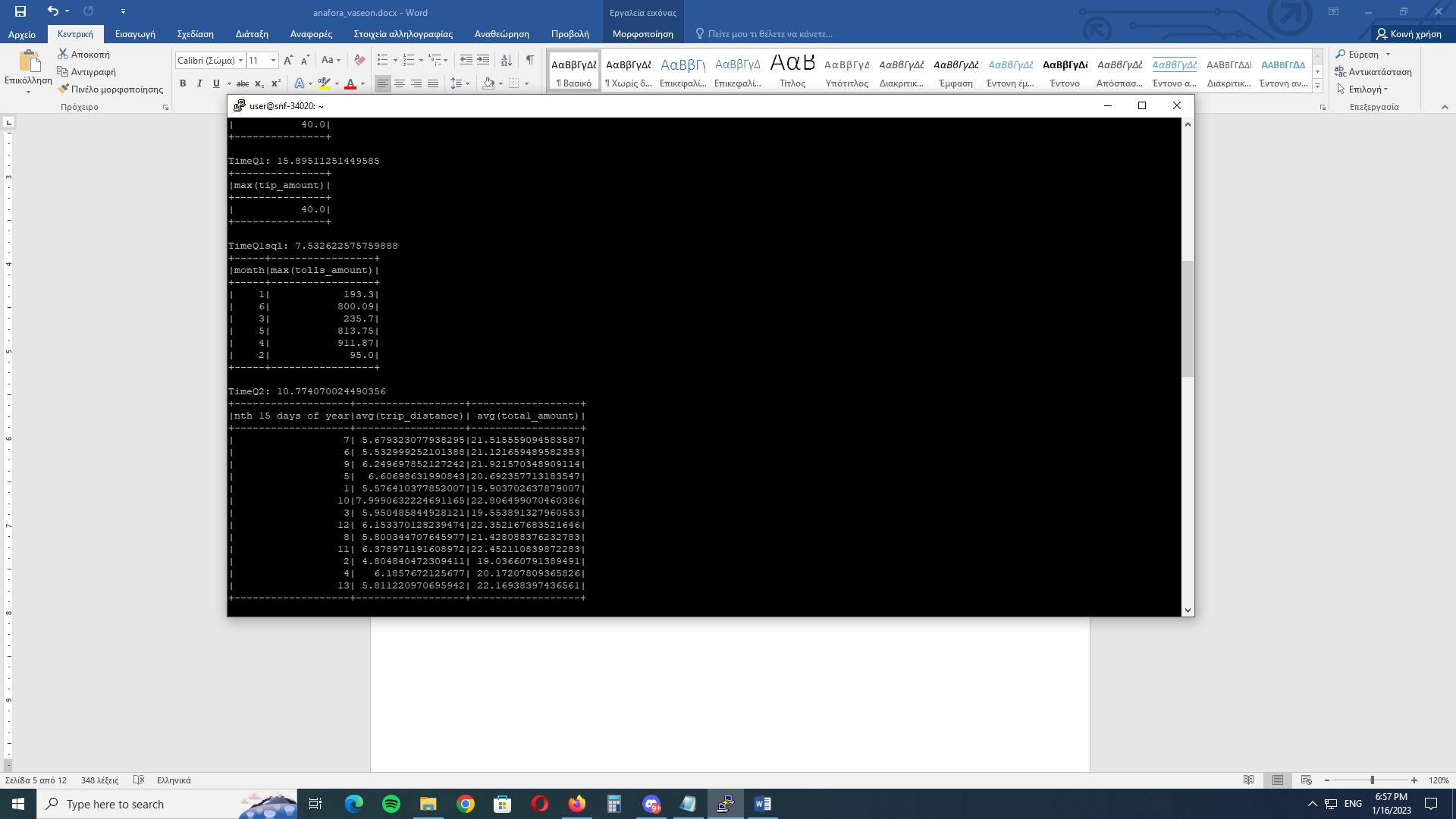


Εκτελούμε το αρχείο με τα queries μέσω της εντολής: python3.8 queriesfinal.py και βλέπουμε τα παρακάτω αποτελέσματα με τους αντίστοιχους χρόνους για όλα τα queries.

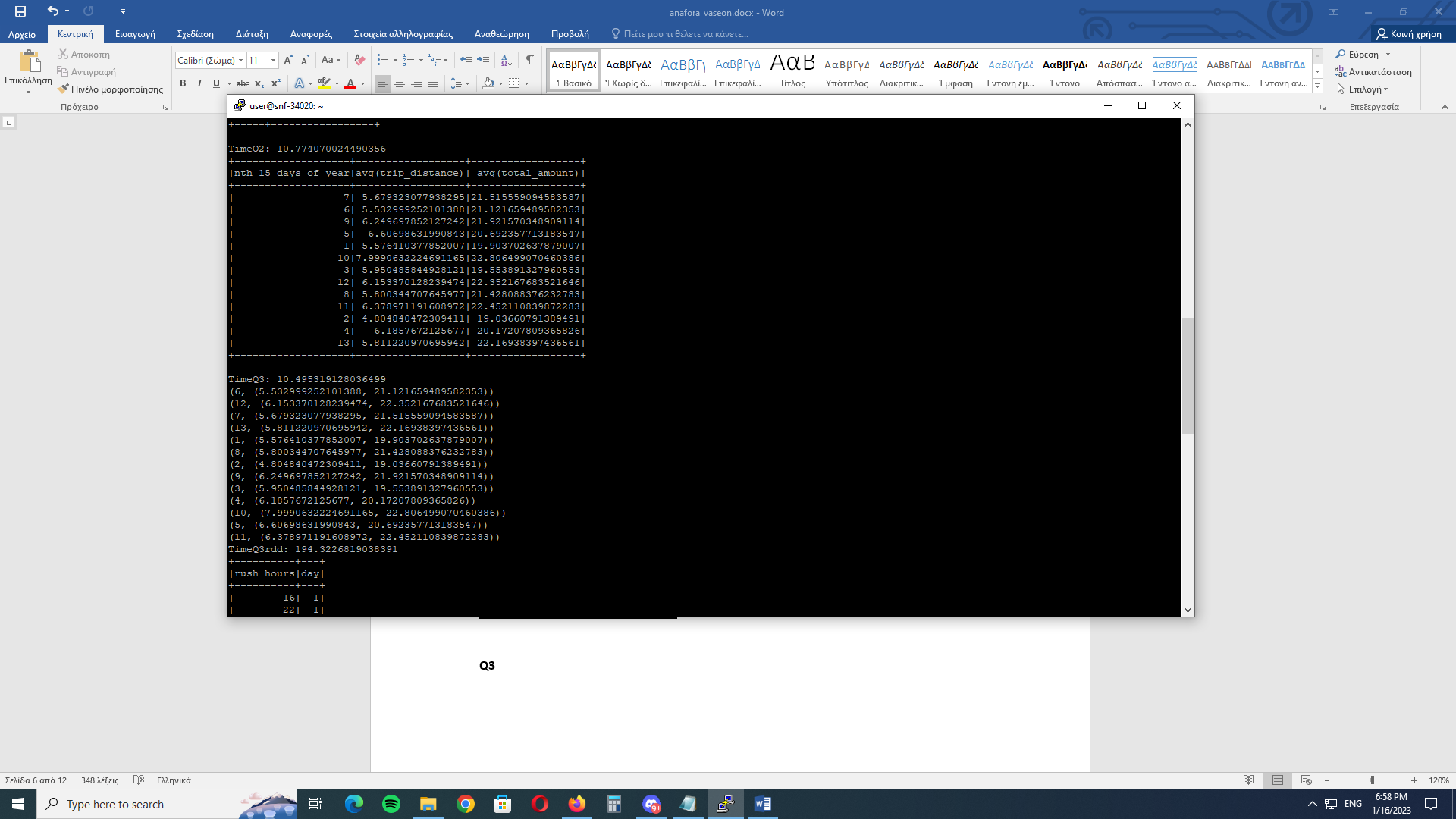
Για 2 Workers:

**Q1**  


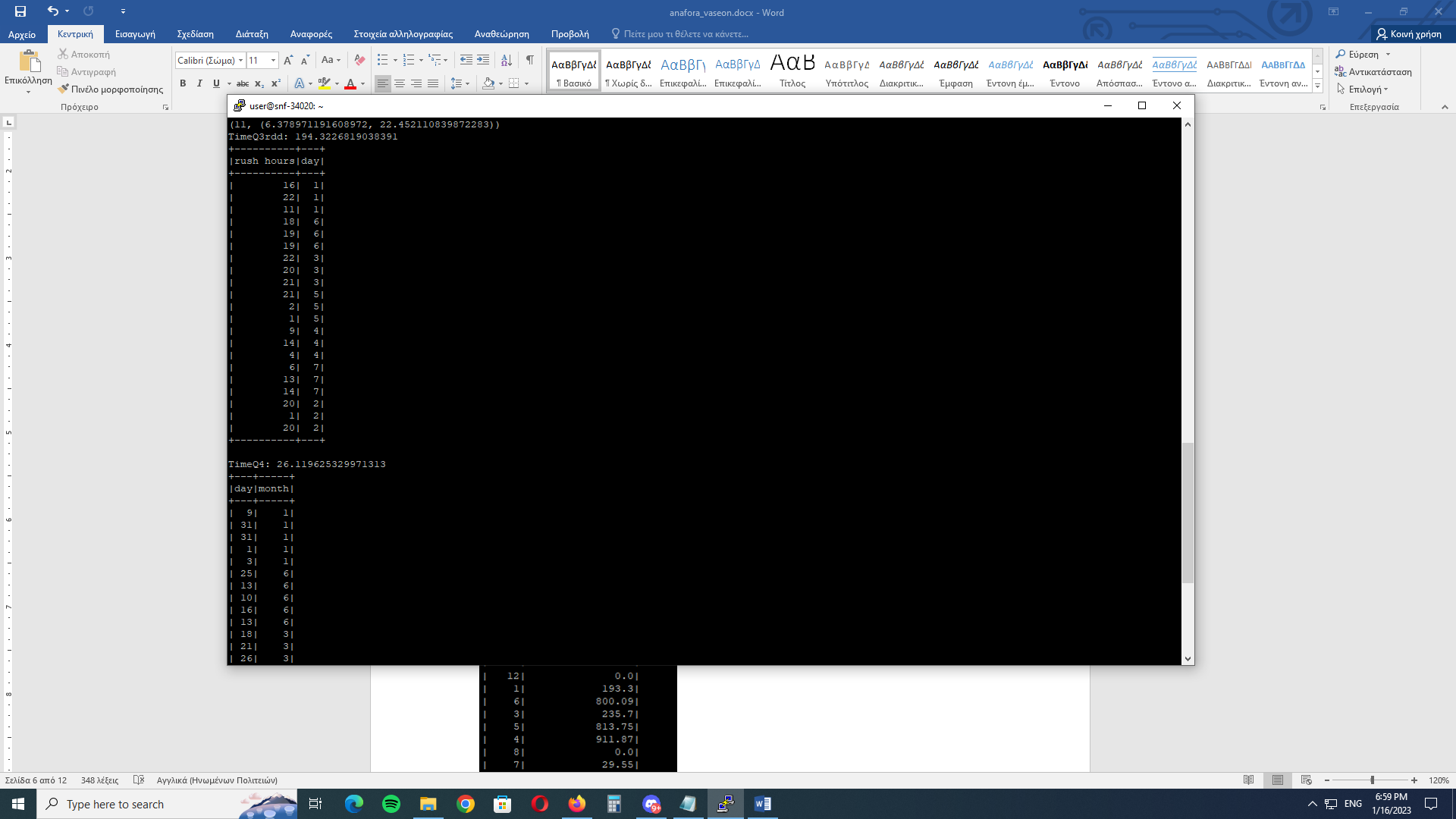
**Q2**



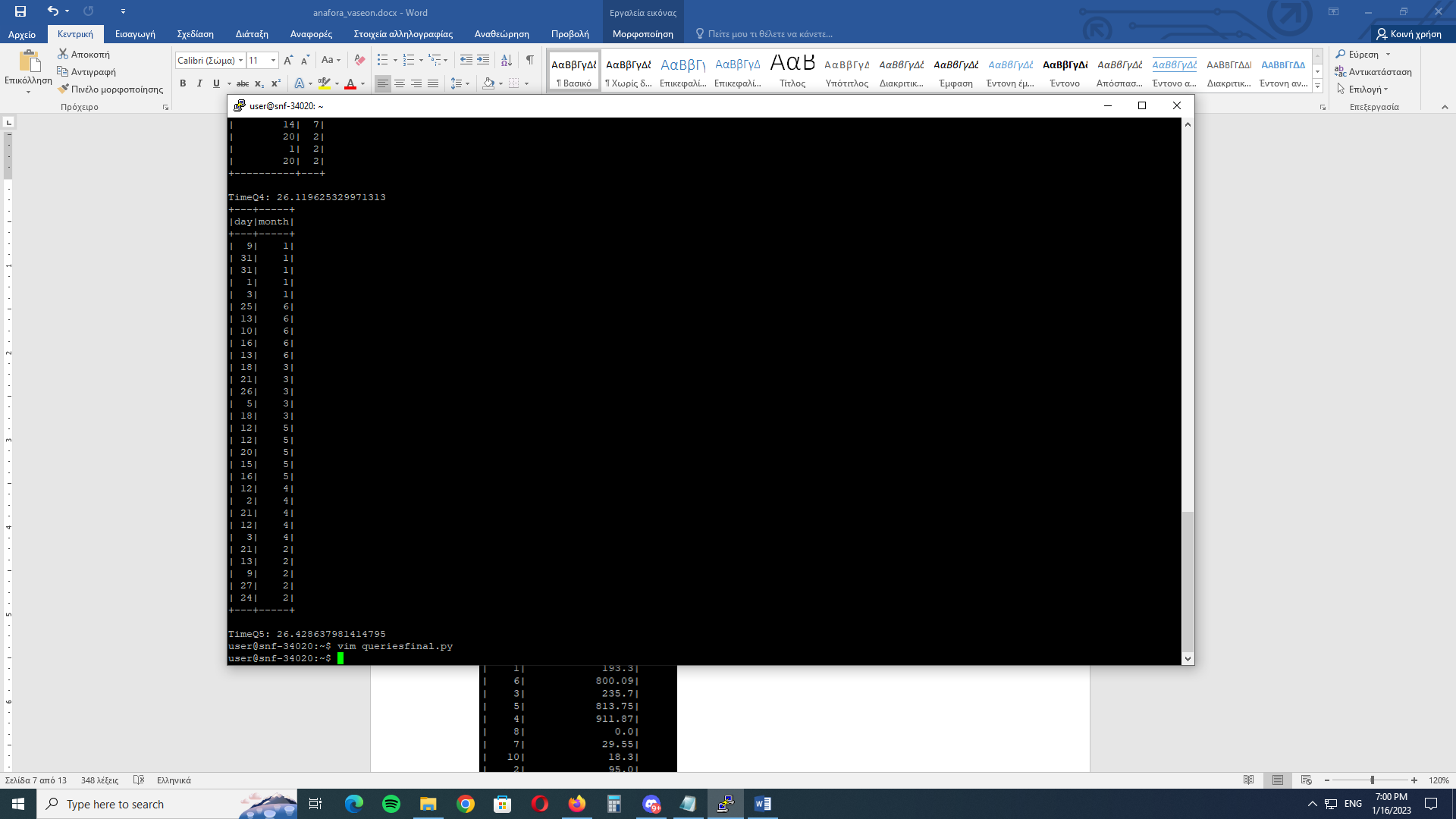
**Q3**



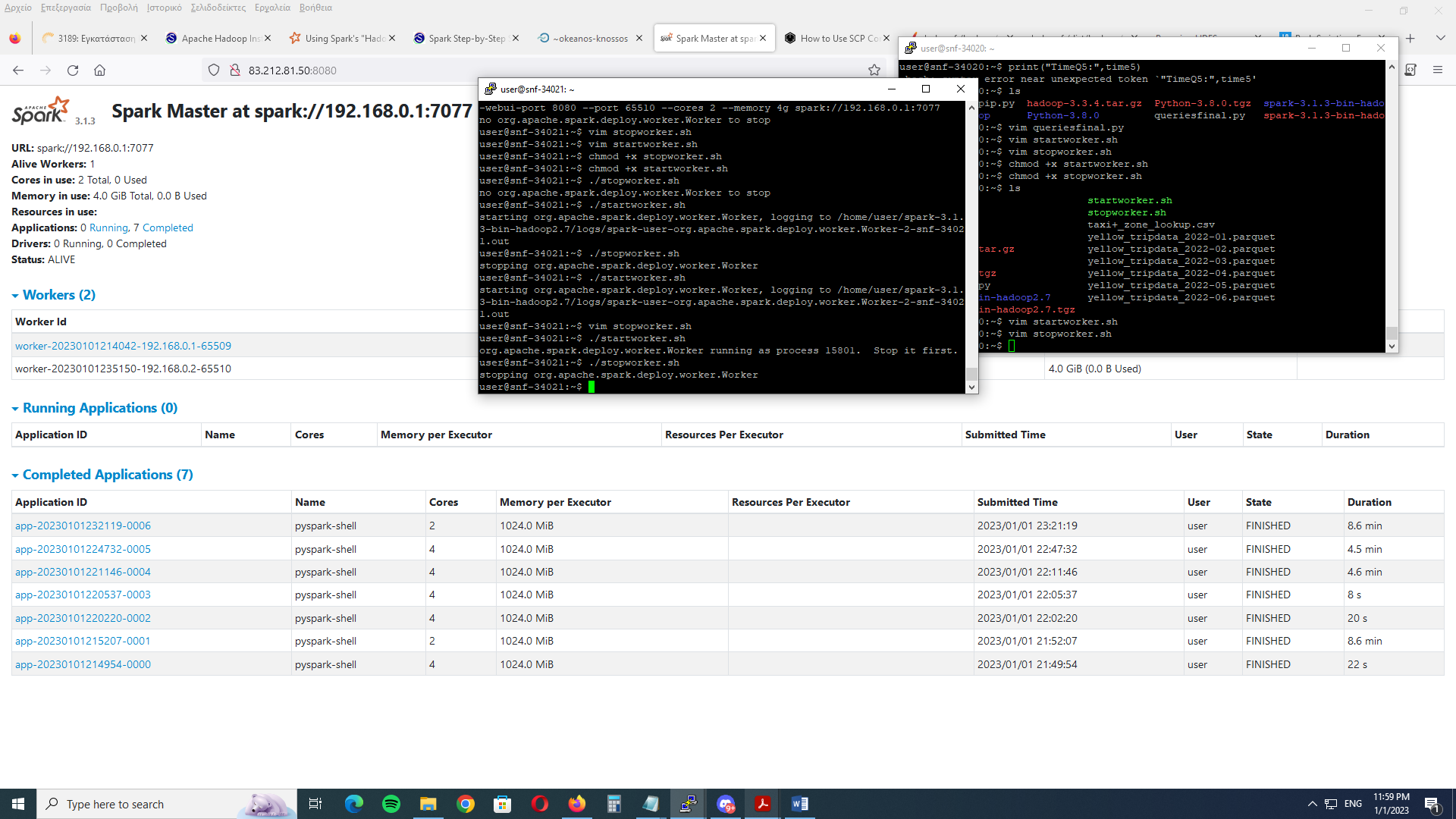
**Q4**

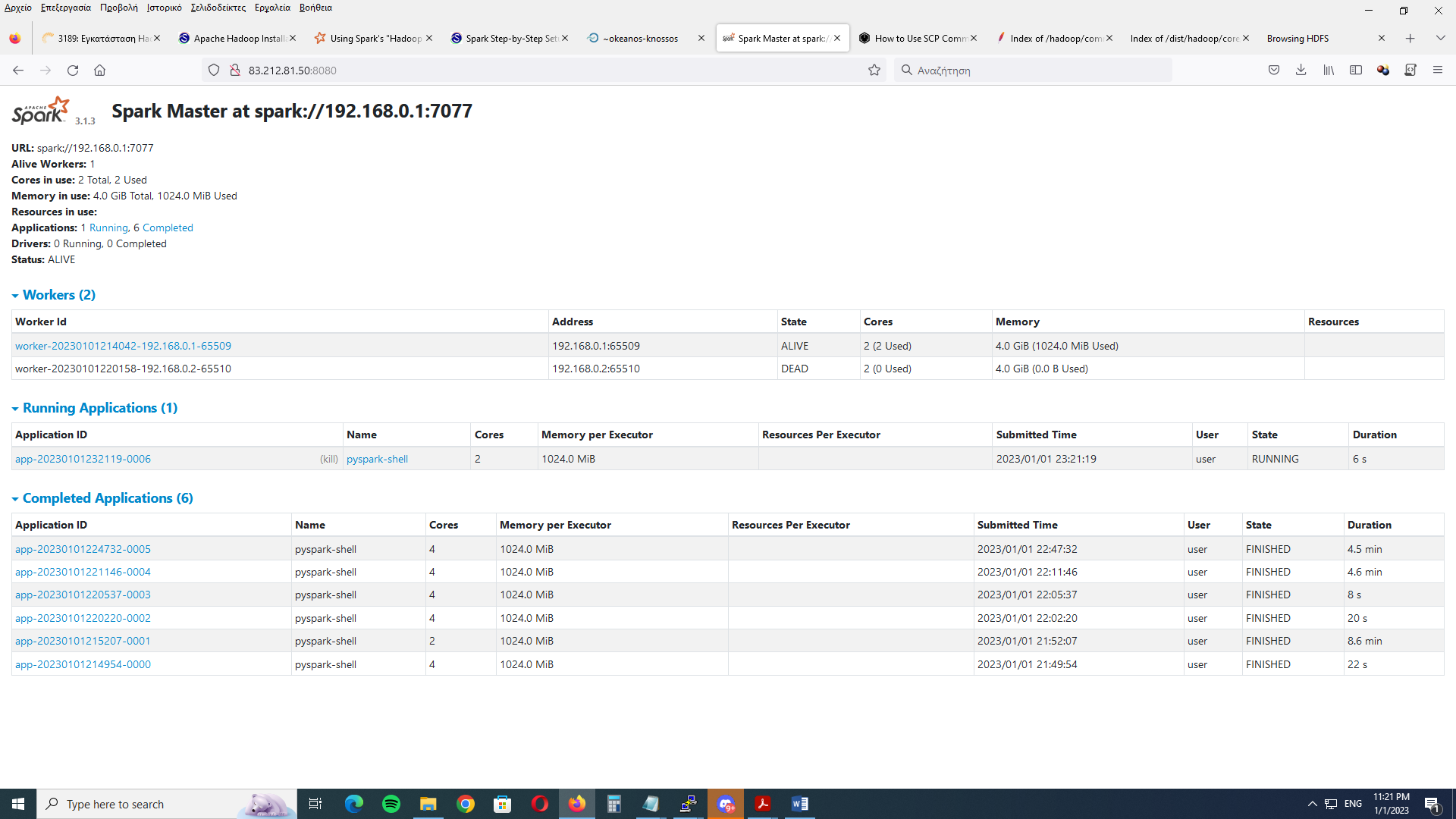


**Q5**



Στη συνέχεια εκτελούμε το script: stopworker.sh που δημιουργήσαμε στο slave node για να απενεργοποιήσουμε τον ένα worker στο spark.

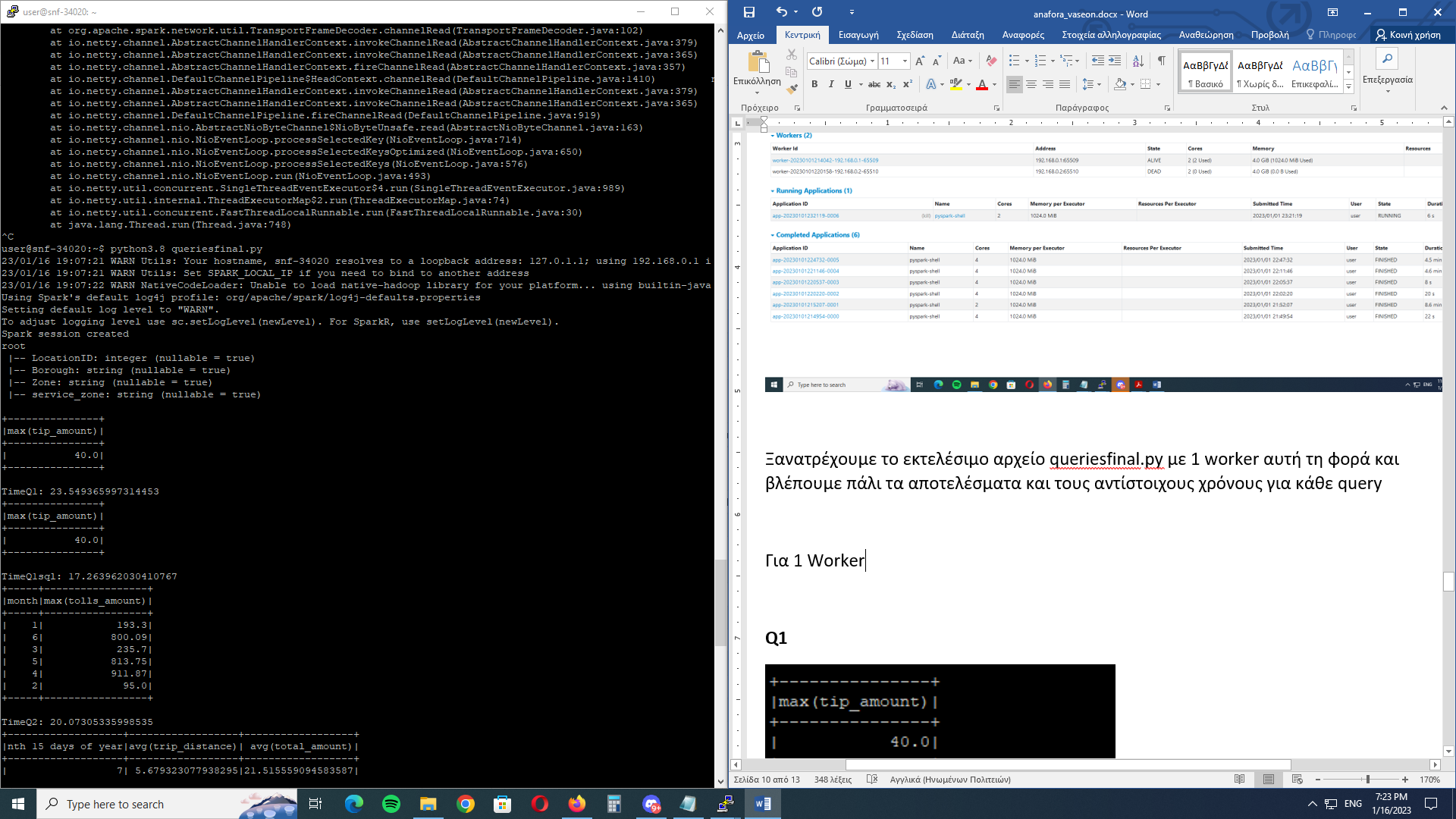




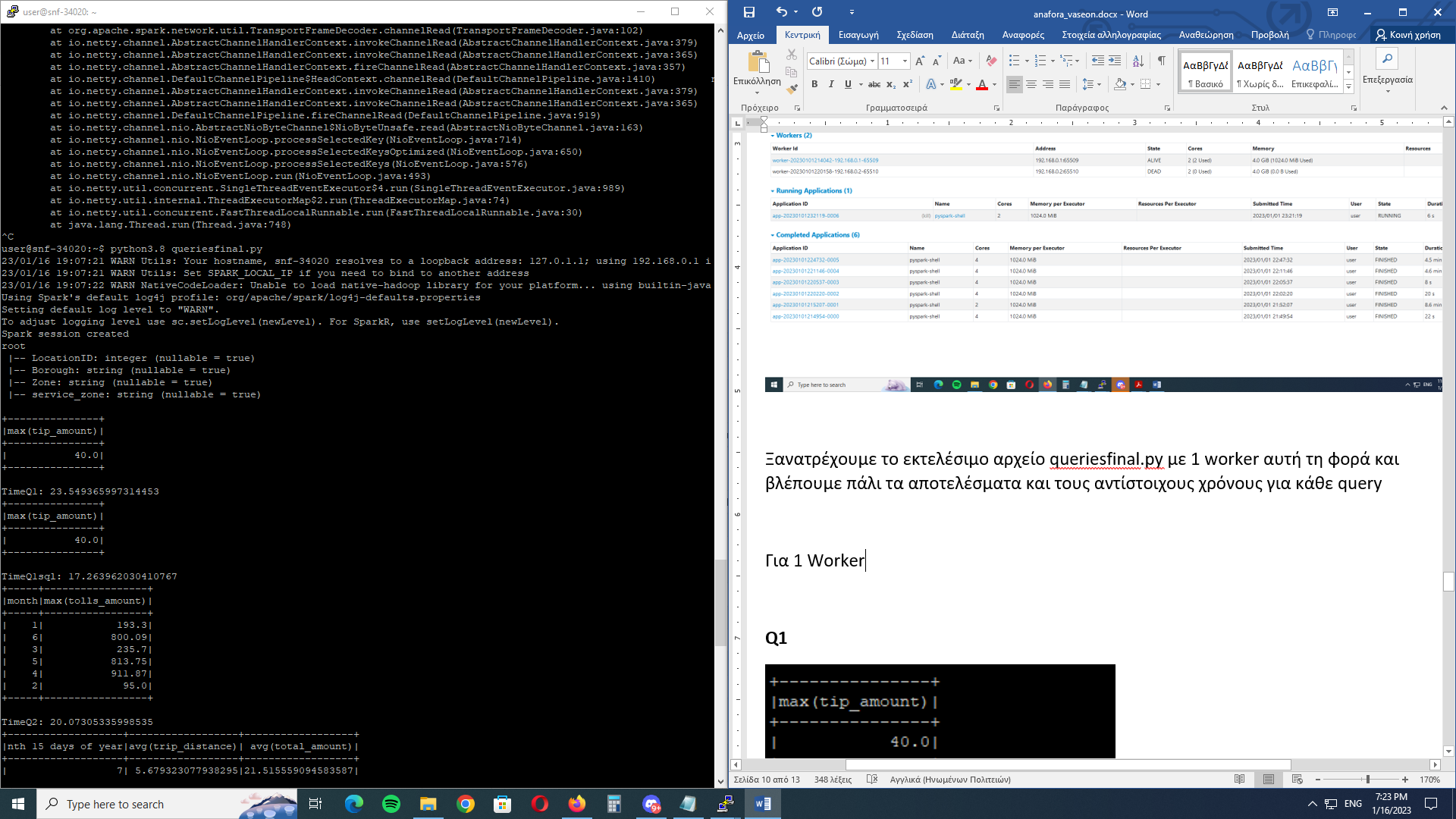
Ξανατρέχουμε το εκτελέσιμο αρχείο queriesfinal.py με 1 worker αυτή τη φορά και βλέπουμε πάλι τα αποτελέσματα και τους αντίστοιχους χρόνους για κάθε query

Για 1 Worker:

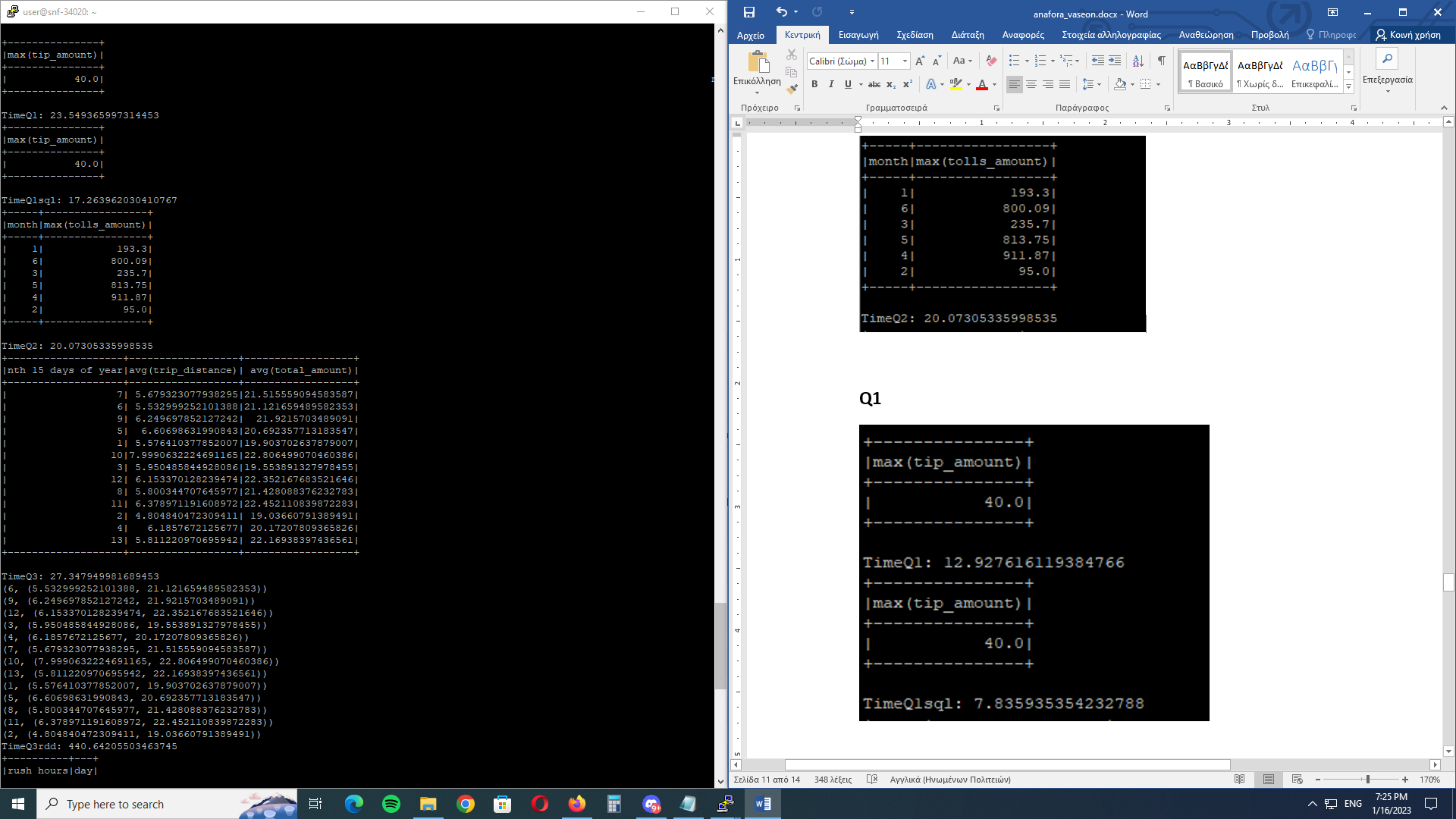
**Q1**



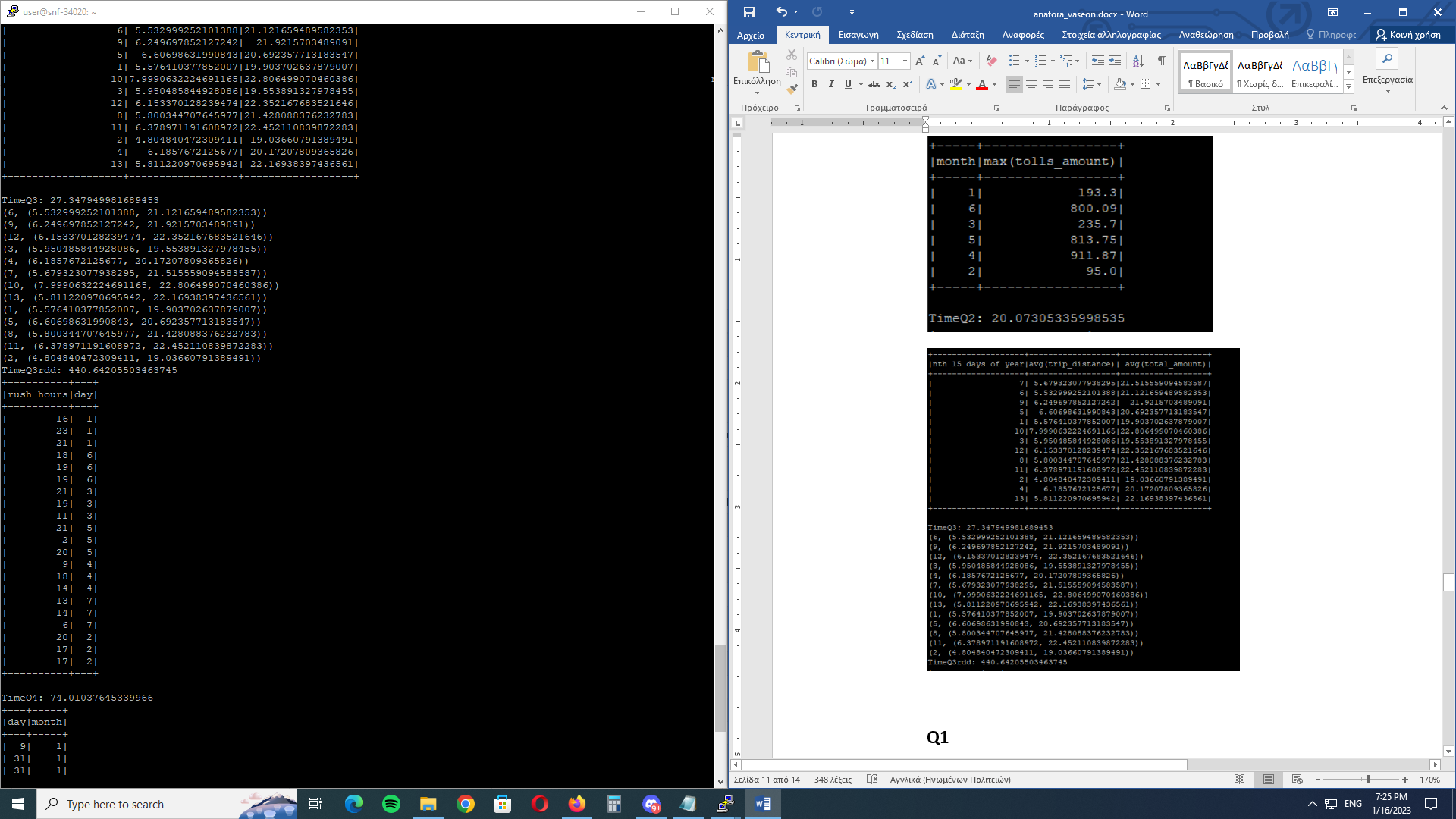
**Q2**



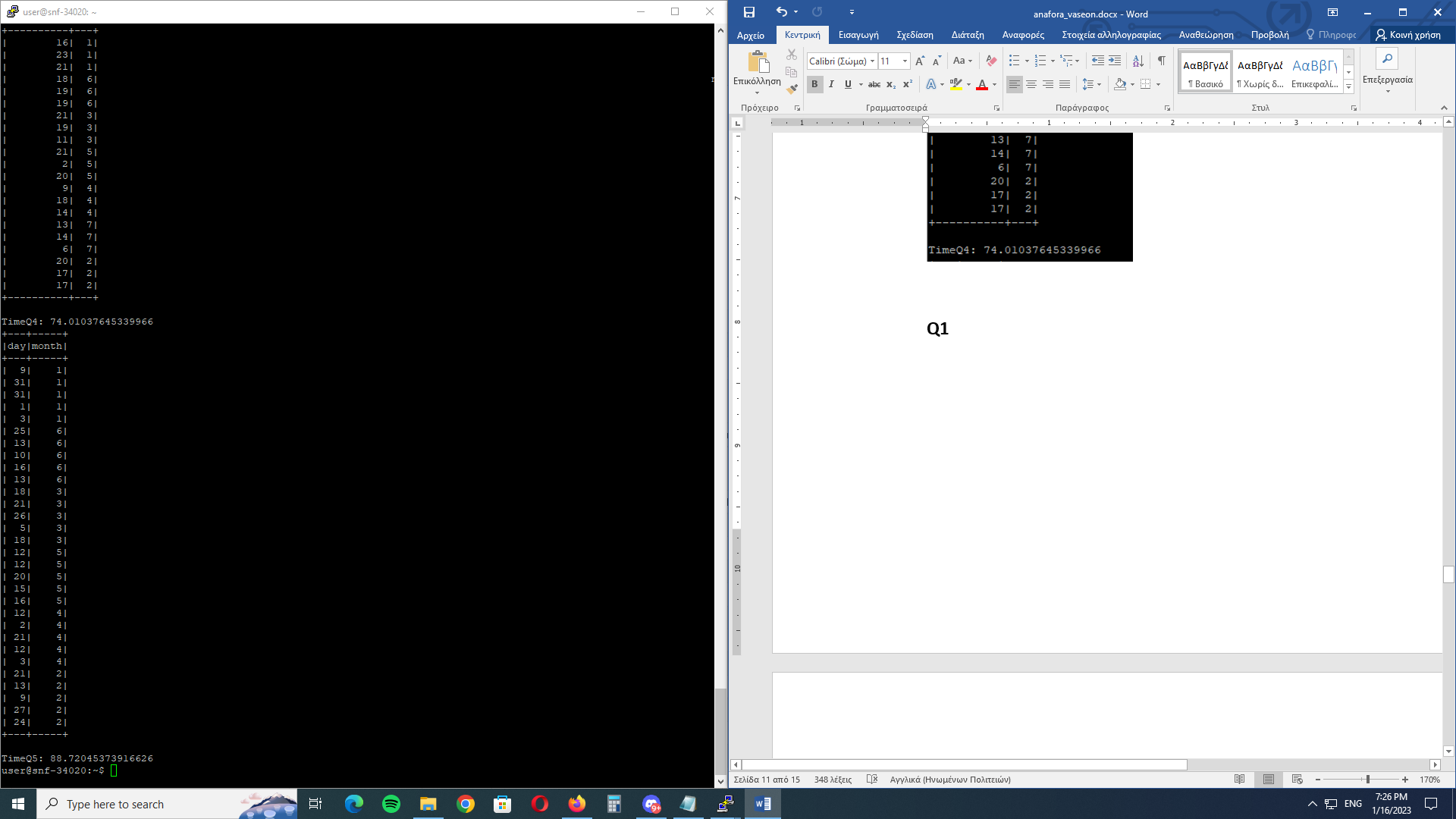
**Q3**



**Q4**



**Q5**



Συνοψίζουμε τα αποτελέσματα με τους χρόνους για κάθε query για έναν και δύο workers αντίστοιχα

|  |  |  |
| --- | --- | --- |
| Time (sec) | One worker | Two workers |
| Q1 | 23.54 | 15.89 |
| Q1sql | 17.26 | 7.53 |
| Q2 | 20.07 | 10.77 |
| Q3 | 27.34 | 10.49 |
| Q3rdd | 440.64 | 194.32 |
| Q4 | 74.01 | 26.11 |
| Q5 | 88.72 | 26.42 |

Βλέπουμε ότι οι χρόνοι μειώνονται σημαντικά με 2 workers γεγονός που αποδεικνύει ότι το setup έχει γίνει σωστά και αποδεικνύει την βελτίωση της επίδοσης μέσω της αύξησης των workers στο apache spark. Ειδικά στο query με rdd η βελτίωση είναι σημαντική.