Big Data

1η Εργασία

Αμανατίδης Ιωάννης - ics23028
Μαυρουδής Δημήτριος - ics23109

Γενικές Πληροφορίες

Περιβάλλον εκτέλεσης:

- CPU: AMD Ryzen 7 5800H with Radeon Graphics
- GPU: Nvidia GeForce GTX 1650 με 4GB μνήμη.
- RAM: 16 GB
- Python Version: Python 3.12.12

Στην εργασία χρησιμοποιήθηκαν το ChatGPT και το Gemini ως υποστηρικτικά εργαλεία για την επεξήγηση κάποιων εντολών και λειτουργιών όπως το combineByKey, το Pandas DataFrame κτλ.

Θέμα 1

Output

Top 10 Origin Airports with Highest Average Departure Delay:
DFW: 11.98
JFK: 11.20
SEA: 9.56
ORD: 9.35
LAS: 9.11
CLT: 8.82
MCO: 8.69
BOS: 8.63
DEN: 8.44
MIA: 8.26

Πίνακας χρόνων εκτέλεσης

Εκτέλεση	**Συνολικός χρόνος**	**Χρόνος πρώτου action**
1	46.02 seconds	4.32 seconds
2	12.32 seconds	1.54 seconds
3	12.30 seconds	1.65 seconds
4	12.39 seconds	1.52 seconds

5	12.44 seconds	1.45 seconds

Max: Εκτέλεση 1 - 46.02 seconds
Min: Εκτέλεση 3 - 12.30 seconds

Εκτέλεση	**Συνολικός χρόνος**	**Χρόνος πρώτου action**
2	12.32 seconds	1.54 seconds
4	12.39 seconds	1.52 seconds
5	12.44 seconds	1.45 seconds
Μέσος όρος	12.38 seconds	1.50 seconds

Θέμα 2

Output

Οι 10 διαδρομές με τη μεγαλύτερη μέση καθυστέρηση αναχώρησης:

```
+--------------+------------+------------------+
|ORIGIN_AIRPORT|DEST_AIRPORT|    avg(DEP_DELAY)|
+--------------+------------+------------------+
|           DFW|         JFK|              23.3|
|           JFK|         LAS|              22.8|
|           MIA|         JFK|              20.5|
|           MCO|         ORD|19.818181818181817|
|           MCO|         LAX|19.142857142857142|
|           BOS|         SEA|18.857142857142858|
|           DFW|         ATL|            18.625|
|           BOS|         PHX|18.181818181818183|
|           JFK|         MIA|              18.0|
|           DEN|         SFO|17.857142857142858|
+--------------+------------+------------------+
```

Πίνακας χρόνων εκτέλεσης

Εκτέλεση	Συνολικός χρόνος	Χρόνος πρώτου action
1	61.07 seconds	5.10 seconds
2	14.56 seconds	1.46 seconds
3	15.00 seconds	1.10 seconds
4	13.35 seconds	1.16 seconds
5	13.20 seconds	0.63 seconds

Max: Εκτέλεση 1 - 61.07 seconds
Min: Εκτέλεση 5 - 13.20 seconds

Εκτέλεση	Συνολικός χρόνος	Χρόνος πρώτου action
2	14.56 seconds	1.46 seconds
3	15.00 seconds	1.10 seconds
4	13.35 seconds	1.16 seconds
Μέσος όρος	14.30 seconds	1.24 seconds

Θέμα 3

Στην εργασία χρησιμοποιήθηκαν δύο υλοποιήσεις: Spark RDD API και Spark DataFrame API. Το DataFrame υπερισχύει ως προς την ευκολία υλοποίησης και εκφραστικότητας κώδικα καθώς έχει το schema, αυτόματη μετατροπή τύπων και έτοιμες λειτουργίες όπως το groupBy. Επίσης ο κώδικας είναι πιο ευανάγνωστος και σύντομος και μοιάζει στην SQL σε αρκετά σημεία, κάτι που τον κάνει σχετικά πιο κατανοητό. Από την άλλη, στο RDD ο κώδικας δεν είναι τόσο κατανοητός καθώς ακολουθεί ένα πιο χαμηλό επίπεδο και κάποια πράγματα, όπως και η μετατροπή τύπων, πρέπει να γίνονται χειροκίνητα.

Όσον αφορά τον χρόνο εκτέλεσης, το αναμενόμενο θα ήταν το DataFrame να είναι γρηγορότερο λόγω του Catalyst Optimizer που εξασφαλίζει αυτόματο optimization. Στην συγκεκριμένη περίπτωση βέβαια, δεν υπήρχε μεγάλη απόκλιση στους χρόνους εκτέλεσης μεταξύ των δύο υλοποιήσεων και μάλιστα το RDD είχε μικρότερο μέσο χρόνο.

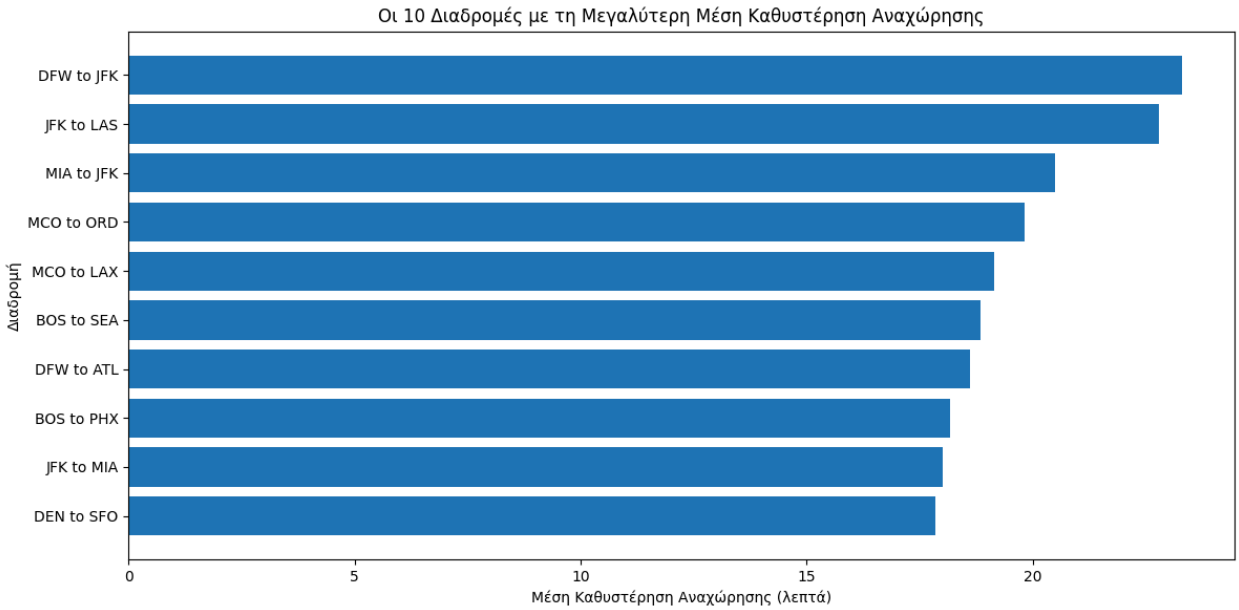
Ίσως ευθύνεται και το γεγονός ότι στο RDD ο υπολογισμός αφορούσε μόνο μέση καθυστέρηση άνα αεροδρόμιο αναχώρησης ενώ στο DataFrame ο υπολογισμός αφορούσε μέση καθυστέρηση ανά διαδρομή (δηλαδή και origin και destination). Λαμβάνοντας υπόψη αυτό το κριτήριο, θα μπορούσαμε να πούμε ότι το DataFrame είναι τελικά γρηγορότερο, βλέποντας πόσο μικρή είναι η απόκλιση στον συνολικό χρόνο, παρά την πιο απαιτητική εργασία που είχε να κάνει, σε σχέση με το RDD.

Σίγουρα, οι δυνατότητες βελτιστοποίησης είναι περισσότερες στο DataFrame καθώς, όπως αναφέρθηκε και πριν, το optimization γίνεται αυτόματα μέσω των Catalyst Optimizer και Tungsten Engine, ενώ στο RDD πρέπει να γίνεται χειροκίνητα.

Υλοποίηση	**Μέσος συνολικός χρόνος**	**Μέσος χρόνος πρώτου action**
Spark RDD API	12.38 seconds	1.50 seconds
Spark DataFrame API	14.30 seconds	1.24 seconds

Λαμβάνοντας υπόψη όλα τα παραπάνω, καταλληλότερη για την επεξεργασία μεγάλων συνόλων δεδομένων φαίνεται να είναι η μέθοδος του Spark DataFrame καθώς με το αυτόματο optimization και τις λειτουργίες που παρέχει εξασφαλίζει μεγαλύτερη αποδοτικότητα σε σχέση με τα RDDs.

Θέμα 4

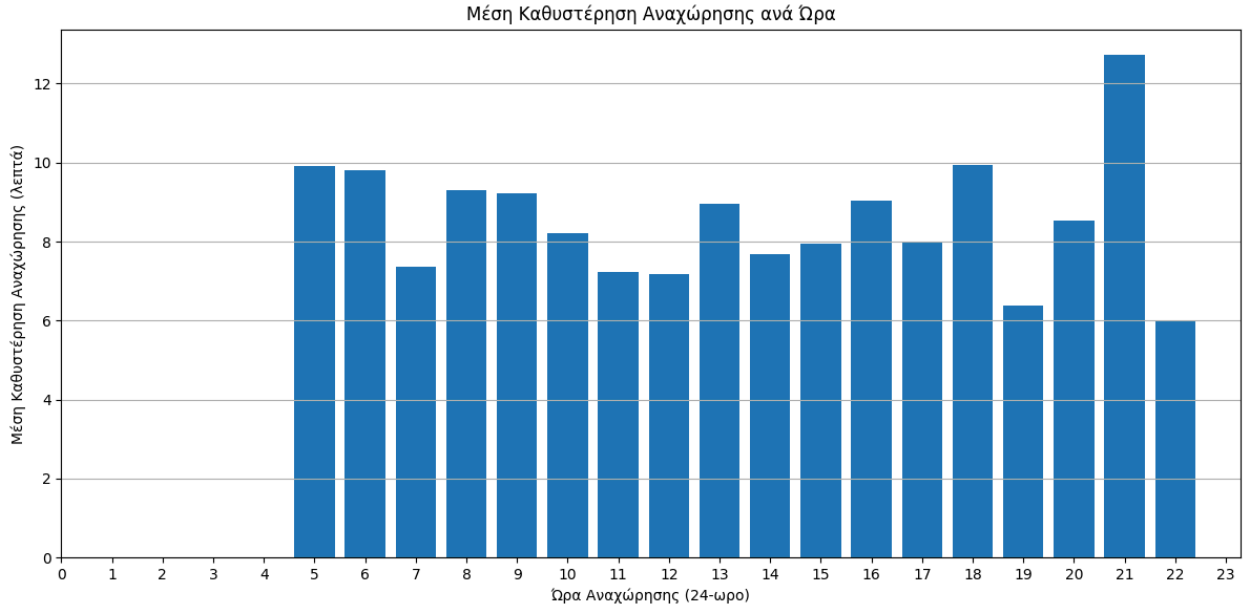


Το γράφημα προέκυψε μετά την μετατροπή του τελικού DataFrame σε Pandas DataFrame. Επιτρέπει την οπτική ανάλυση των αποτελεσμάτων της επεξεργασίας που πραγματοποιήθηκε πάνω στα δεδομένα, κάτι που μπορεί να διευκολύνει σημαντικά την ανάλυση. Για παράδειγμα, στην συγκεκριμένη περίπτωση, μπορούμε να δούμε και να συγκρίνουμε καλύτερα και πιο οργανωμένα τις αποκλίσεις στις καθυστερήσεις μεταξύ αεροδρομίων. Όπως φαίνεται, οι διαδρομές DFW-JFK και JFK-LAS έχουν αρκετά υψηλότερες μέσες καθυστερήσεις, το οποίο μπορεί να οφείλεται σε διάφορους παράγοντες όπως καιρικές συνθήκες ή τεχνικά ζητήματα.

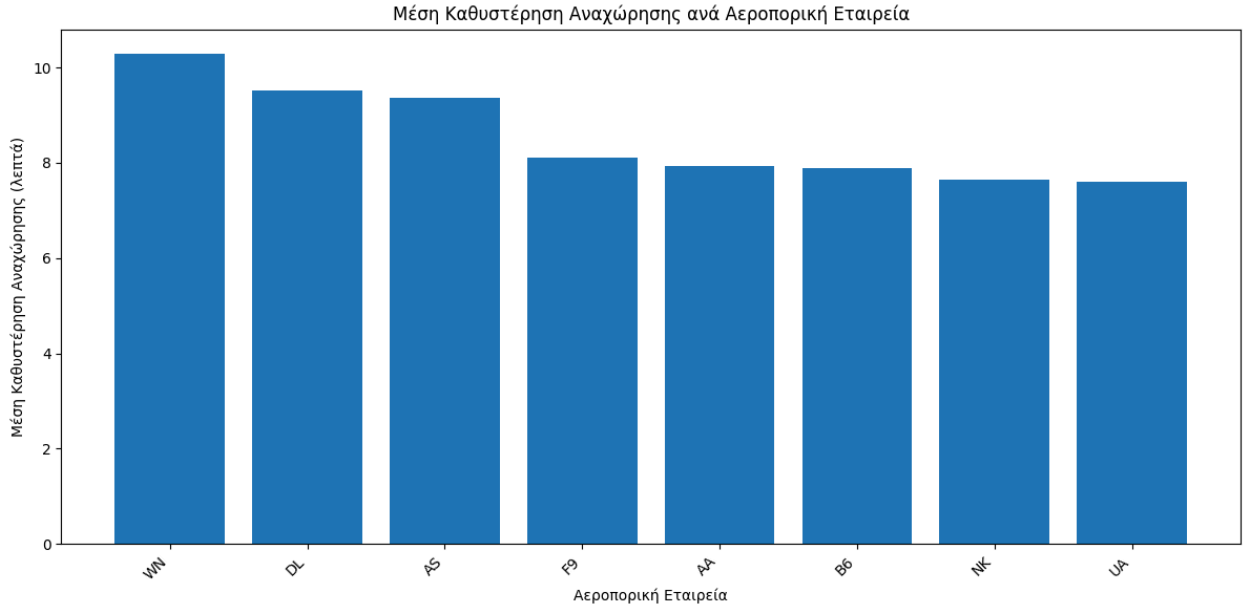
Θέμα 5

Η επέκταση της ανάλυσης έχει ως στόχο να ερευνήσει τις καθυστερήσεις αναχώρησης από διαφορετικές οπτικές γωνίες. Λαμβάνονται υπόψη περισσότερα κριτήρια και αυτό οδηγεί στην κατανόηση παραγόντων που επηρεάζουν τις καθυστερήσεις. Για παράδειγμα, η ανάλυση της μέσης καθυστέρησης ανά ώρα δείχνει τις ώρες που

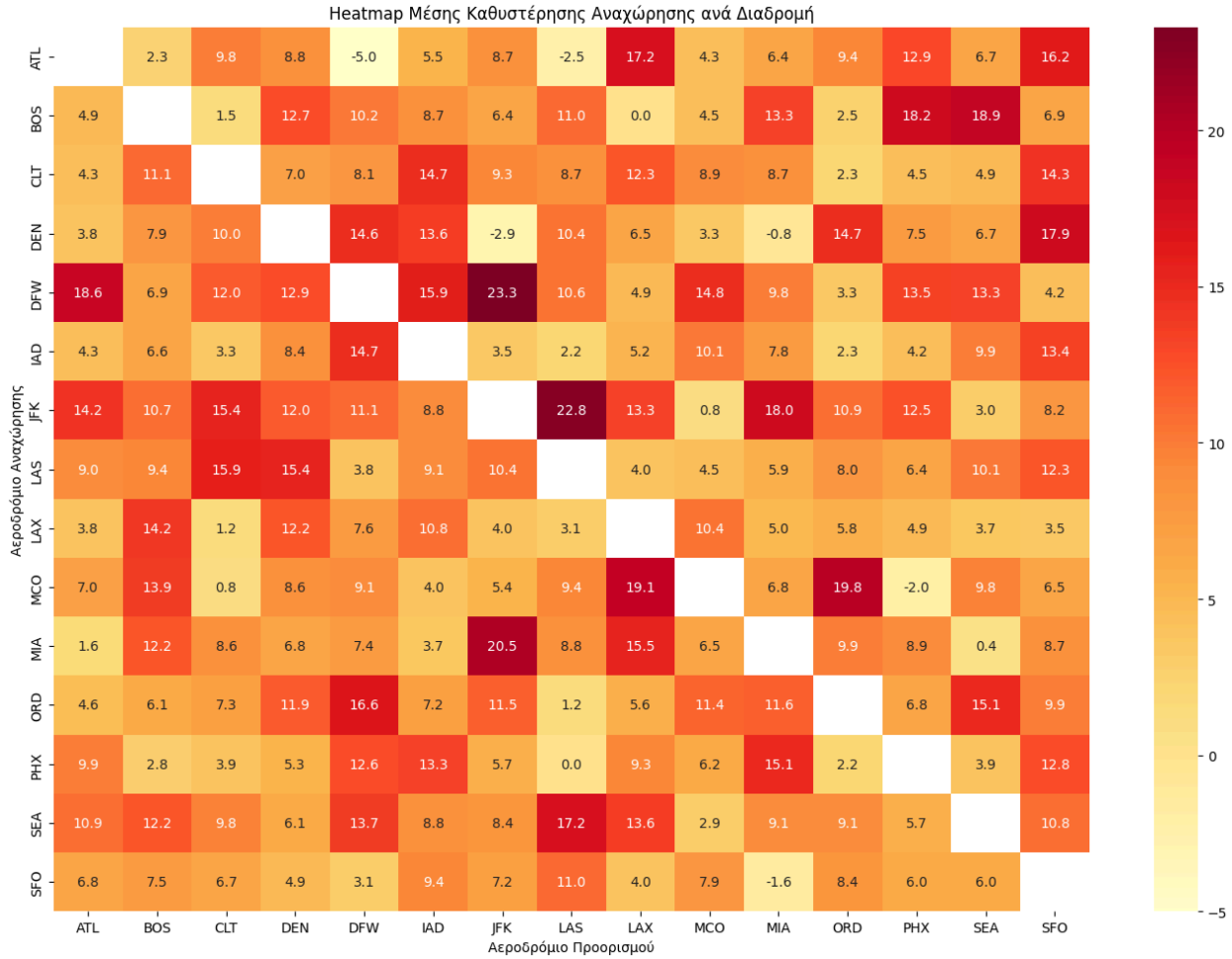
εμφανίζονται οι περισσότερες καθυστερήσεις και η ανάλυση της μέσης καθυστέρησης ανά εταιρεία δείχνει τις εταιρείες στις οποίες παρατηρούνται οι περισσότερες καθυστερήσεις. Οπότε υπάρχουν περισσότερες ενδείξεις για τις πιθανές αιτίες, άρα και περισσότερες πιθανότητες για την επίλυση τους.



Στο παραπάνω γράφημα, φαίνεται ότι οι περισσότερες καθυστερήσεις εμφανίζονται στις 21:00 και έπειτα σε απογευματινές ώρες (18:00) ή πολύ πρωινές (5:00 - 6:00).



Στο παραπάνω γράφημα, φαίνεται ότι οι περισσότερες καθυστερήσεις συμβαίνουν σε δρομολόγια της εταιρείας WN. Μετά ακολουθούν οι DL και AS.



Το παραπάνω γράφημα είναι ένα heatmap της μέσης καθυστέρησης αναχώρησης ανά διαδρομή. Δείχνει καλύτερα τις διαδρομές στις οποίες θα πρέπει να δώσουμε βάση, όπως η DF2-JFK με μέση καθυστέρηση 23.3.