

## ▼ Introduction

The aim of this essay is to introduce us to pandas library and to demonstrate how we can apply functions of pandas library to extract insights of the dataset.

The description of the dataset can be found in the following link

(<https://www.kaggle.com/code/abdurrehmankhalid/airplane-dataset-analysis>)

First thing first, we have to upload our dataframe. I upload the dataframe from my google drive. You can upload the dataframe also from your pc

```
import pandas as pd                # import the main library
from google.colab import drive    # import the dataframe
drive.mount("/content/gdrive")
```

Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive



```
data=pd.read_excel("/content/gdrive/MyDrive/Pandas Delaited Flights/DelayedFlights.xlsx")
```

## ▼ First question:

Find and display the exact number of rows in each column that contains nas.

Πρώτη ερώτηση

να βρείτε (αν υπάρχουν) και να εμφανίσετε το πλήθος των κενών γραμμών σε κάθε στήλη του dataset

```
data.isnull().sum()
```

```
Unnamed: 0      0
Year            0
Month           0
DayOfMonth      0
DayOfWeek       0
DepTime         0
```

```

CRSDepTime      0
ArrTime         3896
CRSArrTime      0
UniqueCarrier   0
FlightNum       0
TailNum         4
ActualElapsedTime 3896
CRSElapsedTime  157
AirTime         3896
ArrDelay        3896
DepDelay        0
Origin          0
Dest            0
Distance        0
TaxiIn          3896
TaxiOut         0
Cancelled       0
CancellationCode 0
Diverted        0
CarrierDelay    362841
WeatherDelay    362841
NASDelay        362841
SecurityDelay   362841
LateAircraftDelay 362841
new             3896
dtype: int64

```

```
data.info()
```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 31 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   Unnamed: 0            1048575 non-null int64
 1   Year                  1048575 non-null int64
 2   Month                 1048575 non-null int64
 3   DayofMonth            1048575 non-null int64
 4   DayOfWeek             1048575 non-null int64
 5   DepTime               1048575 non-null float64
 6   CRSDepTime            1048575 non-null int64
 7   ArrTime               1044679 non-null float64
 8   CRSArrTime            1048575 non-null int64
 9   UniqueCarrier         1048575 non-null object
10   FlightNum             1048575 non-null int64
11   TailNum               1048571 non-null object
12   ActualElapsedTime     1044679 non-null float64
13   CRSElapsedTime        1048418 non-null float64
14   AirTime               1044679 non-null float64
15   ArrDelay              1044679 non-null float64
16   DepDelay              1048575 non-null float64
17   Origin                1048575 non-null object
18   Dest                  1048575 non-null object
19   Distance              1048575 non-null int64
20   TaxiIn                1044679 non-null float64
21   TaxiOut               1048575 non-null float64
22   Cancelled             1048575 non-null int64
23   CancellationCode      1048575 non-null object
24   Diverted              1048575 non-null int64

```

```
25  CarrierDelay      685734 non-null    float64
26  WeatherDelay      685734 non-null    float64
27  NASDelay          685734 non-null    float64
28  SecurityDelay     685734 non-null    float64
29  LateAircraftDelay 685734 non-null    float64
30  new               1044679 non-null    float64
dtypes: float64(15), int64(11), object(5)
memory usage: 248.0+ MB
```

## ▼ Second Question

Calculate and display in which day and in which month were noted the most delays

### Ερώτηση 2:

να υπολογίσετε και να εμφανίσετε ποια ημέρα σε ποιον μήνα σημειώθηκαν οι περισσότερες καθυστερήσεις πτήσεων

```
data["Month"].nunique()
d_2=data[["Month"]].value_counts()
d_2
```

```
Month
3      200842
2      189534
1      183527
6      166336
4      155264
5      153072
dtype: int64
```

```
d_2=data[["Month", "ArrDelay", "DayofMonth"]].groupby(["Month", "DayofMonth"]).max().reset_index()
d_2.sort_values(by='ArrDelay', ascending=False)
```

	Month	DayofMonth	ArrDelay
33	2	3	2461.0
...	...	...	...



```
d_2=data[["Month","ArrDelay","DayofMonth"]].groupby(["Month","DayofMonth"]).max().reset_index()
```

	Month	DayofMonth	ArrDelay
33	2	3	2461.0
100	4	10	2453.0
126	5	6	1951.0
171	6	20	1707.0
117	4	27	1542.0
...	...	...	...
128	5	8	478.0
38	2	8	474.0
140	5	20	456.0
84	3	25	448.0



The answer is in second month on 3 day 2461 flights were delayed

### ▼ Third Question

Calculate and display the daily mean of delays for each summer month

Ερώτηση 3:

να υπολογίσετε και να εμφανίσετε τον ημερήσιο μέσο όρο καθυστερήσεων για καθέναν από τους θερινούς μήνες του 2008

```
#print(data["Month"],type)
#"ArrDelay"
```

```
d_3=data.loc[(data["Year"]==2008) & (data["Month"]==6),["Month","ArrDelay","DayofMonth"]].
d_3
```

	Month	ArrDelay
DayofMonth		
8	6.0	63.704009
4	6.0	59.005208
22	6.0	57.745626
29	6.0	55.294398
26	6.0	54.224555
16	6.0	53.228567
6	6.0	50.792295
3	6.0	50.416815
10	6.0	49.778672
13	6.0	48.819565
9	6.0	47.340510
15	6.0	47.300329
5	6.0	46.668798
23	6.0	46.451584
19	6.0	46.348048
27	6.0	45.115582
14	6.0	44.656481
7	6.0	43.175140
28	6.0	43.050468
18	6.0	40.803148
25	6.0	39.595063
17	6.0	39.385833
21	6.0	38.924731
11	6.0	38.640430



## ▼ Fourth 4:

Calculate and display the name of the air transport company which had the biggest number of canceled flights type B

### Ερώτηση 4

να υπολογίσετε και να εμφανίσετε το όνομα της αεροπορικής εταιρίας που είχε το μεγαλύτερο πλήθος κωδικών ακύρωσης τύπου B

```
print(data["CancellationCode"],type)
```

```
#data["CancellationCode"].count_values()
data["CancellationCode"].nunique()
#data["CancellationCode"]=="B"
```

```
0      N
1      N
2      N
3      N
4      N
..
1048570  N
1048571  N
1048572  N
1048573  N
1048574  N
Name: CancellationCode, Length: 1048575, dtype: object <class 'type'>
1
```

```
print(data["UniqueCarrier"],type)
```

```
data["UniqueCarrier"].nunique()
```

```
0      WN
1      WN
2      WN
3      WN
4      WN
..
1048570  AA
1048571  AA
1048572  AA
1048573  AA
1048574  AA
Name: UniqueCarrier, Length: 1048575, dtype: object <class 'type'>
20
```

There is no cancelation code type B


## ▼ Fifth 5

Find the number of flights with biggest number of delays

### Ερώτηση 5

να βρείτε τους κωδικούς των πτήσεων με τον μεγαλύτερο αριθμό καθυστερήσεων

```
d_5=data.loc[:,["FlightNum","ArrDelay"]].groupby("FlightNum").sum().reset_index().sort_val
d_5.head(10)
```

	FlightNum	ArrDelay	
<b>376</b>	378	39044.0	
<b>48</b>	50	37402.0	
<b>34</b>	36	36718.0	
<b>319</b>	321	36429.0	
<b>508</b>	510	35726.0	
<b>73</b>	75	34889.0	
<b>332</b>	334	32868.0	
<b>682</b>	685	32060.0	

## ▼ Sixth 6:

Find and display the name of the biggest destination with the most delays

### Ερώτηση 6

να βρείτε και να υπολογίσετε το όνομα του μεγαλύτερου σε απόσταση προορισμού με τις περισσότερες καθυστερήσεις

```
d_6=data.loc[:,["Origin","Dest","Distance","ArrDelay"]].groupby(["Origin","Dest"]).agg({"A
d_6
```

	Origin	Dest	ArrDelay	
0	ORD	LGA	147849.0	
1	LGA	ORD	134776.0	
2	LAX	SFO	129523.0	
3	SFO	LAX	113563.0	
4	ORD	EWR	102937.0	
...	...	...	...	
5069	DAB	CLT	-22.0	
5070	LNK	ABE	-26.0	
5071	OAK	HNL	-35.0	
5072	SMF	OGG	-123.0	

## ▼ Seventh 7:

Find and display the destinations that had the biggest delay

να βρείτε και να εμφανίσετε τους προορισμούς που είχαν την μεγαλύτερη καθυστέρηση (πτήσεις που εκτελέστηκαν)

```
d_7=data.loc[data["CancellationCode"]=="N",["Dest","ActualElapsedTime"]].groupby("Dest").as_d_7
```





## ▼ Nineth Question

Calculate how many type A flight cancellations occurred on 13th day of each month

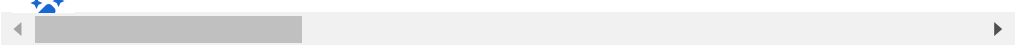
Ερώτηση 9:

να υπολογίσετε πόσες ακυρώσεις πτήσεων τύπου A σημειώθηκαν την 13η ημέρα κάθε μήνα

```
5          00          1151441.0
```

```
d_9=data.loc[(data["DayofMonth"]==13) & (data["CancellationCode"]=="A"),:]
d_9
```

```
Unnamed: 0  Year  Month  DayofMonth  DayOfWeek  DepTime  CRSDepTime
0 rows x 31 columns
```



None

## ▼ Tenth question

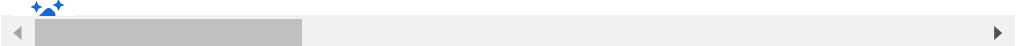
calculate and display the average delay of flights operated from 10 to 19 April 2008

Ερώτηση 10:

υπολογίσετε και να εμφανίσετε την μέση καθυστέρηση πτήσεων που εκτελέστηκαν από την 10η μέχρι την 19 Απριλίου 2008

```
d_10=data.loc[(data["DayofMonth"]==10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19) & (data["Month"]==4) & (data["Year"]==2008)]
d_10
# We dont have any flights
```

```
Unnamed: 0  Year  Month  DayofMonth  DayOfWeek  DepTime  CRSDepTime
0 rows x 31 columns
```



## ▼ Eleventh question

calculate and display the month with the longest delay due to security checks during the hours 06.00-14.00

να υπολογίσετε και να εμφανίσετε τον μήνα που σημειώθηκε η μεγαλύτερη καθυστέρηση που οφειλόταν σε έλεγχους ασφαλείας κατά τις ώρες 06.00-14.00

```
data["SecurityDelay"].sum()
#data["DepTime"]
```

63968.0

```
import numpy as np
d_11=data.loc[(data["DepTime"]>=600) & (data["DepTime"]<=1400),["Month","SecurityDelay","I
d_11
#"np.nanmean"
```

	Month	SecurityDelay
0	3	7486.0
1	1	5555.0
2	2	4719.0
3	5	4287.0
4	6	3217.0

```
import numpy as np
d_11=data.loc[(data["DepTime"]>=600) & (data["DepTime"]<=1400),["Month","SecurityDelay","I
.groupby("Month").agg({"SecurityDelay":"sum"}).sort_values(by="SecurityDelay",ascending=False)
d_11
#"np.nanmean"
```

	Month	SecurityDelay
0	3	7486.0
1	1	5555.0
2	2	4719.0
3	5	4287.0
4	6	3217.0

## ▼ Twelveth question.

Calculate and print which code of flight had the first and the tenth day of 6 month of 2008 the


### Ερώτηση 12:

να υπολογίσετε και να εμφανίσετε ποιος κωδικός πτήσης(αριθμός πτήσης) είχε την πρώτη και δέκατη μέρα έκτου μήνα του 2008 την μεγαλύτερη προ του αναμενόμενου χρόνου άφιξη στον προορισμό της

```
# Πρώτος τρόπος
#First way
d_12=data.loc[(data["DayofMonth"]==1) | (data["DayofMonth"]==10) & (data["Month"]==6) & (c
d_12["new"]=d_12["ArrTime"]-d_12["CRSArrTime"]
d_12.groupby("FlightNum").agg({"new":"sum"})
d_12
```

	DayofMonth	Month	ArrTime	CRSArrTime	FlightNum	new
<b>30743</b>	1	1	1303.0	1245	2679	58.
<b>30751</b>	1	1	854.0	821	1	33.
<b>30754</b>	1	1	6.0	2342	10	-2336.
<b>30782</b>	1	1	1219.0	1146	103	73.
<b>30792</b>	1	1	1240.0	1223	105	17.
...	...	...	...	...	...	.
<b>1048449</b>	1	6	1430.0	1350	1481	80.
<b>1048517</b>	1	6	2356.0	2330	1488	26.
<b>1048547</b>	1	6	1720.0	1640	1494	80.

```
d_12=data.loc[(data["DayofMonth"]==1) | (data["DayofMonth"]==10) & (data["Month"]==6) & (c
new=d_12.groupby("FlightNum").apply(lambda d_12: (d_12["ArrTime"] - d_12["CRSArrTime"])).r
#sort_values(by="0",ascending=False).reset_index()
new
#d_12
```

	FlightNum	level_1	0	
<b>0</b>	1	30751	33.0	
<b>1</b>	1	66587	19.0	
<b>2</b>	1	168251	40.0	
<b>3</b>	1	168334	100.0	
<b>4</b>	1	175614	-3.0	
...	...	...	...	
<b>38827</b>	7826	220845	52.0	
<b>38828</b>	7826	414762	66.0	
<b>38829</b>	7828	33252	117.0	
<b>38830</b>	7828	220857	92.0	

# Δεύτερος τρόπος

```
def subtract(x):
```

```
    # x is a DataFrame of group values
```

```
    x["new"]=d_12["ArrTime"]-d_12["CRSArrTime"]
```

```
    return x
```

# modified the question and we take the first 10 days

```
d_12=data.loc[(data["DayofMonth"]==1|2|3|4|5|6|7|8|9|10) & (data["Month"]==6) & (data["Arr
```

```
new=d_12.groupby("FlightNum").apply(subtract).reset_index()
```

```
ok=new.sort_values(by="new",ascending=False)
```

```
#sort_values(by="0",ascending=False).reset_index()
```

```
ok
```

	index	DayofMonth	Month	ArrTime	CRSArrTime	FlightNum
<b>617</b>	1031732	15	6	1227.0	1228	1785

```
d_12=data.loc[(data["DayofMonth"]==1|2|3|4|5|6|7|8|9|10) & (data["Month"]==6) & (data["Arr
.assign(num=lambda x: x["ArrTime"]-x["CRSArrTime"]).groupby("FlightNum").agg({"num":"sum"})
```

```
# den trexei gt to Arrtime exei nas
d_12
```

	FlightNum	num
<b>0</b>	9	-1.0
<b>1</b>	1006	-1.0
<b>2</b>	1785	-1.0
<b>3</b>	1688	-1.0
<b>4</b>	1679	-1.0
...	...	...
<b>629</b>	206	-2357.0
<b>630</b>	6642	-2358.0
<b>631</b>	24	-2367.0
<b>632</b>	60	-4256.0


## ▼ Thirteenth question

find and display flights that were diverted but completed and the total time taken

### Ερώτηση 13:

να βρείτε και να εμφανίσετε τις πτήσεις που εκτράπηκαν από την πορεία τους αλλά ολοκληρώθηκαν καθώς και τον συνολικό χρόνο που απαιτήθηκε

```
d_14=data.loc[(data["Diverted"]==1) & (data["CancellationCode"]=="N"),["Diverted","CancellationCode"]
.assign(num=lambda x: x["CRSElapsedTime"]+x["DepDelay"])\
.groupby(["FlightNum","TailNum"]).agg({"num":"sum"}).sort_values(by="num",ascending=False).
d_14
```

	FlightNum	TailNum	num	
0	1181	N3AXAA	1230.0	
1	3338	N904AE	1224.0	
2	1282	N833MH	1220.0	
3	1273	N830MH	1209.0	
4	3602	N689EC	995.0	
...	...	...	...	
3827	5730	80269E	0.0	
3828	3729	86969E	0.0	
3829	9741	91629E	0.0	
3830	2780	N710BR	-2.0	

## ▼ Fourteenth

which month had the largest standard deviation in delays ("most unpredictable month"). The difference between the planned and the actual flight execution time should be considered as a deviation


## Ερώτηση 14:

ποιος μήνας είχε την μεγαλύτερη τυπική απόκλιση σε καθυστερήσεις ("πιο απρόβλεπτος μήνας"). Ως απόκλιση να θεωρηθεί η διαφορά ανάμεσα στον προγραμματισμένο και τον πραγματικό χρόνο εκτέλεσης της πτήσης

```
import statistics
from statistics import stdev
data["new"]=np.subtract(data["CRSElapsedTime"],data["ActualElapsedTime"])
data["new"].describe()
data["new"].std()
```

16.9215187679791

```
d_15=data.loc[(data["new"]>0),["Month","new"]].groupby("Month").agg({"new":["std","mean"],s
#sort_values(by="std",ascending=False).reset_index()
d_15
#groupby("Month")
```



	Month	new				
		std	mean	sum	min	max
0	1	7.620670	10.352972	1115336.0	1.0	90.0
1	2	7.428301	10.028041	1087531.0	1.0	101.0
2	3	7.529717	10.081771	1213946.0	1.0	151.0
3	4	7.433828	10.004471	924163.0	1.0	76.0
4	5	7.457911	10.026340	937162.0	1.0	181.0

[Colab paid products](#) - [Cancel contracts here](#)

 0s    completed at 3:11 PM

