1 Exercici 1

Descarrega el data set Airlines Delay: Airline on-time statistics and delay causes i carrega'l a les dades que conté, i queda't únicament amb les columnes que consideris rellevants.

In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
df = pd.read_csv("DelayedFlights.csv", index_col = 0)
```

```
C:\Users\Nuria\anaconda3\lib\site-packages\numpy\lib\arraysetops.py:583: FutureWarning: elementwise comparison
stead, but in the future will perform elementwise comparison
  mask |= (ar1 == a)
```

In [3]:

```
df.head(10)
```

	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueC
0	2008	1	3	4	2003.0	1955	2211.0	2225	WN
1	2008	1	3	4	754.0	735	1002.0	1000	WN
2	2008	1	3	4	628.0	620	804.0	750	WN
4	2008	1	3	4	1829.0	1755	1959.0	1925	WN
5	2008	1	3	4	1940.0	1915	2121.0	2110	WN
6	2008	1	3	4	1937.0	1830	2037.0	1940	WN
10	2008	1	3	4	706.0	700	916.0	915	WN
11	2008	1	3	4	1644.0	1510	1845.0	1725	WN
15	2008	1	3	4	1029.0	1020	1021.0	1010	WN
16	2008	1	3	4	1452.0	1425	1640.0	1625	WN

10 rows × 29 columns

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1936758 entries, 0 to 7009727
Data columns (total 29 columns):
 #   Column             Dtype  
---  ------             -----  
 0   Year               int64  
 1   Month              int64  
 2   DayofMonth         int64  
 3   DayOfWeek          int64  
 4   DepTime            float64
 5   CRSDepTime         int64  
 6   ArrTime            float64
 7   CRSArrTime         int64  
 8   UniqueCarrier      object 
 9   FlightNum          int64  
 10  TailNum            object 
 11  ActualElapsedTime  float64
 12  CRSElapsedTime     float64
 13  AirTime            float64
 14  ArrDelay           float64
 15  DepDelay           float64
 16  Origin             object 
 17  Dest               object 
 18  Distance           int64  
 19  TaxiIn             float64
 20  TaxiOut            float64
 21  Cancelled          int64  
 22  CancellationCode   object 
 23  Diverted           int64  
 24  CarrierDelay       float64
 25  WeatherDelay       float64
 26  NASDelay           float64
 27  SecurityDelay      float64
 28  LateAircraftDelay  float64
dtypes: float64(14), int64(10), object(5)
memory usage: 443.3+ MB
```

Al dataset no hi ha informació sobre què signifiquen els noms de les columnes, però hi ha u
[https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/HG7NV7]
(https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/HG7NV7%5D).
explicació de què representa cada columna.

In [5]:

```
df = df.drop(columns = ["CancellationCode", "Origin", "Dest", "Year", "Cancelled"]
#Year eliminat perque es sempre 2008
#CancellationCode i Cancelled perquè estem analitzant només els enraderiments
#Origin i Dest no son necessaris per l'analisis
```

In [6]:

```
df.head()
```

	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier
0	1	3	4	2003.0	1955	2211.0	2225	WN
1	1	3	4	754.0	735	1002.0	1000	WN
2	1	3	4	628.0	620	804.0	750	WN
4	1	3	4	1829.0	1755	1959.0	1925	WN
5	1	3	4	1940.0	1915	2121.0	2110	WN

5 rows × 24 columns

In [7]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1936758 entries, 0 to 7009727
Data columns (total 24 columns):
 #   Column             Dtype  
---  ------             -----  
 0   Month              int64  
 1   DayofMonth         int64  
 2   DayOfWeek          int64  
 3   DepTime            float64
 4   CRSDepTime         int64  
 5   ArrTime            float64
 6   CRSArrTime         int64  
 7   UniqueCarrier      object 
 8   FlightNum          int64  
 9   TailNum            object 
 10  ActualElapsedTime  float64
 11  CRSElapsedTime     float64
 12  AirTime            float64
 13  ArrDelay           float64
 14  DepDelay           float64
 15  Distance           int64  
 16  TaxiIn             float64
 17  TaxiOut            float64
 18  Diverted           int64  
 19  CarrierDelay       float64
 20  WeatherDelay       float64
 21  NASDelay           float64
 22  SecurityDelay      float64
 23  LateAircraftDelay  float64
dtypes: float64(14), int64(8), object(2)
memory usage: 369.4+ MB
```

2 Exercici 2

Fes un informe complet del data set:

- Resumeix estadísticament les columnes d'interès

In [8]:

```
print("Mitjana ActualElapsedTime:", df["ActualElapsedTime"].mean())
print("Desviació estàndar ActualElapsedTime:", df["ActualElapsedTime"].std())
print("Mediana ActualElapsedTime:", df["ActualElapsedTime"].median())
print("Moda ActualElapsedTime:", df["ActualElapsedTime"].mode())
print("En minuts")
```

```
Mitjana ActualElapsedTime: 133.30586334268665
Desviació estàndar ActualElapsedTime: 72.06006897518652
Mediana ActualElapsedTime: 116.0
Moda ActualElapsedTime: 0    80.0
dtype: float64
En minuts
```

In [9]:

```
print("Mitjana CRSElapsedTime:", df["CRSElapsedTime"].mean())
print("Desviació estàndar CRSElapsedTime:", df["CRSElapsedTime"].std())
print("Mediana CRSElapsedTime:", df["CRSElapsedTime"].median())
print("Moda CRSElapsedTime:", df["CRSElapsedTime"].mode())
print("En minuts")
```

```
Mitjana CRSElapsedTime: 134.3027440409799
Desviació estàndar CRSElapsedTime: 71.34143888168691
Mediana CRSElapsedTime: 116.0
Moda CRSElapsedTime: 0    75.0
dtype: float64
En minuts
```

In [10]:

```
print("Mitjana AirTime:", df["AirTime"].mean())
print("Desviació estàndar AirTime:", df["AirTime"].std())
print("Mediana AirTime:", df["AirTime"].median())
print("Moda AirTime:", df["AirTime"].mode())
print("En minuts")
```

```
Mitjana AirTime: 108.27714739539228
Desviació estàndar AirTime: 68.6426101386457
Mediana AirTime: 90.0
Moda AirTime: 0    45.0
dtype: float64
En minuts
```

In [11]:

```
print("Mitjana ArrDelay:", df["ArrDelay"].mean())
print("Desviació estàndar ArrDelay:", df["ArrDelay"].std())
print("Mediana ArrDelay:", df["ArrDelay"].median())
print("Moda ArrDelay:", df["ArrDelay"].mode())
print("En minuts")
```

```
Mitjana ArrDelay: 42.19988477321014
Desviació estàndar ArrDelay: 56.78471513743561
Mediana ArrDelay: 24.0
Moda ArrDelay: 0    10.0
dtype: float64
En minuts
```

In [12]:

```
print("Mitjana DepDelay:", df["DepDelay"].mean())
print("Desviació estàndar DepDelay:", df["DepDelay"].std())
print("Mediana DepDelay:", df["DepDelay"].median())
print("Moda DepDelay:", df["DepDelay"].mode())
print("En minuts")
```

```
Mitjana DepDelay: 43.185176464999756
Desviació estàndar DepDelay: 53.40250234363254
Mediana DepDelay: 24.0
Moda DepDelay: 0    6.0
dtype: float64
En minuts
```

In [13]:

```
print("Mitjana Distance:", df["Distance"].mean())
print("Desviació estàndar Distance:", df["Distance"].std())
print("Mediana Distance:", df["Distance"].median())
print("Moda Distance:", df["Distance"].mode())
print("En milles")
```

```
Mitjana Distance: 765.6861590348407
Desviació estàndar Distance: 574.4796530725632
Mediana Distance: 606.0
Moda Distance: 0    337
dtype: int64
En milles
```

- Troba quantes dades faltants hi ha per columna

In [15]:

```
df.isna().sum()
```

```
Month                   0
DayofMonth              0
DayOfWeek               0
DepTime                 0
CRSDepTime              0
ArrTime              7110
CRSArrTime              0
UniqueCarrier           0
FlightNum               0
TailNum                 5
ActualElapsedTime    8387
CRSElapsedTime        198
AirTime              8387
ArrDelay             8387
DepDelay                0
Distance                0
TaxiIn               7110
TaxiOut               455
Diverted                0
CarrierDelay       689270
WeatherDelay       689270
NASDelay           689270
SecurityDelay      689270
LateAircraftDelay  689270
dtype: int64
```

- Crea columnes noves (velocitat mitjana del vol, si ha arribat tard o no...)

In [17]:

```
df["DistanceKm"] = df["Distance"] * 1.60934    #convertir de milles a km
df["AirTime"].replace(0, np.nan, inplace = True)
df["AirTimeH"] = df["AirTime"] / 60    #convertir de minuts a hores
df["FlightSpeed"] = df["DistanceKm"] / df["AirTimeH"]    #crear columna de velocit
print("Velocitat mitjana de vol:", df["FlightSpeed"].mean(), "km/h")
```

```
Velocitat mitjana de vol: 638.8318056975423 km/h
```

In [18]:

```
df["LateLanding"] = df["ArrDelay"] > 0
#si ha arribat tard o no
```

In [19]:

```
df["LateTakeOff"] = df["DepDelay"] > 0
#si ha sortit tard o no
```

In [20]:

```
df["ElapsedTimeDifference"] = df["CRSElapsedTime"] - df["ActualElapsedTime"]
#diferencia entre el temps transcorregut esperat i el real en minuts. Inclou el te
```

In [21]:

```
df["ArrivalDifference"] = df["CRSArrTime"] - df["ArrTime"]
#diferencia entre el temps d'arribada esperat i el real
```

In [22]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1936758 entries, 0 to 7009727
Data columns (total 31 columns):
 #   Column                Dtype  
---  ------                -----  
 0   Month                 int64  
 1   DayofMonth            int64  
 2   DayOfWeek             int64  
 3   DepTime               float64
 4   CRSDepTime            int64  
 5   ArrTime               float64
 6   CRSArrTime            int64  
 7   UniqueCarrier         object 
 8   FlightNum             int64  
 9   TailNum               object 
 10  ActualElapsedTime     float64
 11  CRSElapsedTime        float64
 12  AirTime               float64
 13  ArrDelay              float64
 14  DepDelay              float64
 15  Distance              int64  
 16  TaxiIn                float64
 17  TaxiOut               float64
 18  Diverted              int64  
 19  CarrierDelay          float64
 20  WeatherDelay          float64
 21  NASDelay              float64
 22  SecurityDelay         float64
 23  LateAircraftDelay     float64
 24  DistanceKm            float64
 25  AirTimeH              float64
 26  FlightSpeed           float64
 27  LateLanding           bool    
 28  LateTakeOff           bool    
 29  ElapsedTimeDifference float64
 30  ArrivalDifference     float64
dtypes: bool(2), float64(19), int64(8), object(2)
memory usage: 447.0+ MB
```

- Taula de les aerolínies amb més endarreriments acumulats

In [23]:

```
late = df["LateLanding"] == True
df[late]["UniqueCarrier"].value_counts()
```

```
WN    324717
AA    172197
MQ    130647
UA    123989
OO    121942
DL    100923
XE     94313
CO     83646
US     83262
EV     75170
NW     72395
FL     65008
YV     63289
OH     49104
B6     48177
9E     46896
AS     34179
F9     25708
HA      7199
AQ       654
Name: UniqueCarrier, dtype: int64
```

- Quins són els vols més llargs?

In [38]:

```
df = df.sort_values(by=["Distance"], ascending = False)
print(df.loc[:, "FlightNum"])
```

```
3561206      15
5906279      15
5354393      15
570303       15
4194957      15
           ...
550589     9002
151660     5610
1637250    2009
2547298    4988
4392215    5572
Name: FlightNum, Length: 1936758, dtype: int64
```

- I els més endarrerits?

In [37]:

```
df = df.sort_values(by=["ArrivalDifference"], ascending = False)
print(df.loc[:, "FlightNum"])
```

```
3191777    5497
5734413      36
2365515     649
6935966    1940
5573115    6445
           ...
2927797      65
535448       64
1718865      64
1107749      64
6945761      65
Name: FlightNum, Length: 1936758, dtype: int64
```

- Etc.

- Relació entre endarreriment i mes. Potser hi ha més endarreriments en mesos que la g

In [26]:

```
late = df["LateLanding"] == True
df[late]["Month"].value_counts()
```

```
6     182955
12    182945
3     179833
2     171311
7     164534
1     163801
8     143013
4     137941
5     135421
11     90855
10     87989
9      82817
Name: Month, dtype: int64
```

- Relació entre endarreriment i dia de la setmana.

In [27]:

```
df[late]["DayOfWeek"].value_counts()
#1 és dilluns i 7 es diumenge
```

```
5    291280
1    258998
4    258790
7    254550
3    233500
2    232407
6    193890
Name: DayOfWeek, dtype: int64
```

- Correlació distància i enrederiment d'arribada. Potser a més distancia recorreguda hi ha

In [28]:

```
df["Distance"].corr(df["ArrDelay"])
```

```
-0.02985259624177865
```

- Correlació enrederiment de sortida i d'arribada. Si es surt tard s'arriba tard?

In [29]:

```
df["DepDelay"].corr(df["ArrDelay"])
```

```
0.9529266852026773
```

- Correlació enrederiment de sortida i velocitat de vol. Potser si un vol surt tard, s'intenta

In [30]:

```
df["DepDelay"].corr(df["FlightSpeed"])
```

```
-0.021017359921312054
```

In [39]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1936758 entries, 3561206 to 4392215
Data columns (total 31 columns):
 #   Column                 Dtype  
---  ------                 -----  
 0   Month                  int64  
 1   DayofMonth             int64  
 2   DayOfWeek              int64  
 3   DepTime                float64
 4   CRSDepTime             int64  
 5   ArrTime                float64
 6   CRSArrTime             int64  
 7   UniqueCarrier          object 
 8   FlightNum              int64  
 9   TailNum                object 
 10  ActualElapsedTime      float64
 11  CRSElapsedTime         float64
 12  AirTime                float64
 13  ArrDelay               float64
 14  DepDelay               float64
 15  Distance               int64  
 16  TaxiIn                 float64
 17  TaxiOut                float64
 18  Diverted               int64  
 19  CarrierDelay           float64
 20  WeatherDelay           float64
 21  NASDelay               float64
 22  SecurityDelay          float64
 23  LateAircraftDelay      float64
 24  DistanceKm             float64
 25  AirTimeH               float64
 26  FlightSpeed            float64
 27  LateLanding            bool    
 28  LateTakeOff            bool    
 29  ElapsedTimeDifference  float64
 30  ArrivalDifference      float64
dtypes: bool(2), float64(19), int64(8), object(2)
memory usage: 447.0+ MB
```

In [48]:

```
df.to_csv("DelayedFlightsNet.csv")
```