# S02 T05: Exploració de les dades

Familiaritza't amb les tècniques d'exploració de les dades mitjantçant la estructura de dades, Dataframe amb la llibreria Pandas.

https://www.milantomin.com/2018-u-s-airlines-delay-analysis/

## - Exercici 1

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

```
1 import pandas as pd
2 import seaborn as sns
3 import matplotlib
```

```
1 # Cargo el fichero
2
3 df = pd.read_csv (r'R:\FileHistory\jmmat\DESKTOP-BBPQ0FO\Data\D\Documentos D\99.- borrar\03.- Data Science IT Academy\Airlines Delay\DelayedFlights.csv')
4 df.head(1)
```

| | Unnamed: 0 | Year | Month | DayofMonth | DayOfWeek | DepTime | CRSDepTime | ArrTime | CRSArrTime | UniqueCarrier | ... | TaxiIn | TaxiOut | Cancelled | CancellationCode | Diverted | CarrierDelay | Weathe |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

```
1 # Obtengo informacion del fichero
2 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1936758 entries, 0 to 1936757
Data columns (total 30 columns):
 #   Column             Dtype
---  ------             -----
 0   Unnamed: 0         int64
 1   Year               int64
 2   Month              int64
 3   DayofMonth         int64
 4   DayOfWeek          int64
 5   DepTime            float64
 6   CRSDepTime         int64
 7   ArrTime            float64
 8   CRSArrTime         int64
 9   UniqueCarrier      object
```

```
 10  FlightNum          int64
 11  TailNum            object
 12  ActualElapsedTime  float64
 13  CRSElapsedTime     float64
 14  AirTime            float64
 15  ArrDelay           float64
 16  DepDelay           float64
 17  Origin             object
 18  Dest               object
 19  Distance           int64
 20  TaxiIn             float64
 21  TaxiOut            float64
 22  Cancelled          int64
 23  CancellationCode   object
 24  Diverted           int64
 25  CarrierDelay       float64
 26  WeatherDelay       float64
 27  NASDelay           float64
 28  SecurityDelay      float64
 29  LateAircraftDelay  float64
dtypes: float64(14), int64(11), object(5)
memory usage: 443.3+ MB
```

```
# Creo una lista con el nombre de las columnas.

columnasInteresantes= df.columns
print(columnasInteresantes)
```

```
Index(['Unnamed: 0', 'Year', 'Month', 'DayofMonth', 'DayOfWeek', 'DepTime',
       'CRSDepTime', 'ArrTime', 'CRSArrTime', 'UniqueCarrier', 'FlightNum',
       'TailNum', 'ActualElapsedTime', 'CRSElapsedTime', 'AirTime', 'ArrDelay',
       'DepDelay', 'Origin', 'Dest', 'Distance', 'TaxiIn', 'TaxiOut',
       'Cancelled', 'CancellationCode', 'Diverted', 'CarrierDelay',
       'WeatherDelay', 'NASDelay', 'SecurityDelay', 'LateAircraftDelay'],
      dtype='object')
```

```
# Elimino las columnas que no necesito ahora
borrarColumnas = ['CancellationCode', 'Diverted', 'CarrierDelay',
                  'WeatherDelay', 'NASDelay', 'SecurityDelay', 'LateAircraftDelay', 'CRSArrTime', 'UniqueCarrier', 'FlightNum', 'TaxiIn', 'TaxiOut',
                  'Unnamed: 0', 'DayofMonth', 'TailNum']

df1 = df.drop(borrarColumnas, axis=1)
```

```
df.shape
```

```
(1936758, 30)
```

```
df.head()
```

| | Unnamed: 0 | Year | Month | DayofMonth | DayOfWeek | DepTime | CRSDepTime | ArrTime | CRSArrTime | UniqueCarrier | ... | TaxiIn | TaxiOut | Canc |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 2008 | 1 | 3 | 4 | 2003.0 | 1955 | 2211.0 | 2225 | WN | ... | 4.0 | 8.0 | |
| **1** | 1 | 2008 | 1 | 3 | 4 | 754.0 | 735 | 1002.0 | 1000 | WN | ... | 5.0 | 10.0 | |
| **2** | 2 | 2008 | 1 | 3 | 4 | 628.0 | 620 | 804.0 | 750 | WN | ... | 3.0 | 17.0 | |

```
# Hago listado de las compañías.

dfUniqueCarrier = df.UniqueCarrier.unique()
dfUniqueCarrier
```

```
array(['WN', 'XE', 'YV', 'OH', 'OO', 'UA', 'US', 'DL', 'EV', 'F9', 'FL',
       'HA', 'MQ', 'NW', '9E', 'AA', 'AQ', 'AS', 'B6', 'CO'], dtype=object)
```

## Exercici 2

## Resumeix estadísticament les columnes d'interès

```
# Saco los valores estadisticos más importantes

print('Saco los valores estadisticos más importantes, de las variables numericas')
df.describe()
```

```
Saco los valores estadisticos más importantes, de las variables numericas
```

| | Unnamed: 0 | Year | Month | DayofMonth | DayOfWeek | DepTime | CRSDepTime | ArrTime | CRSArrTime | Fli |
|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 1.936758e+06 | 1936758.0 | 1.936758e+06 | 1.936758e+06 | 1.936758e+06 | 1.936758e+06 | 1.936758e+06 | 1.929648e+06 | 1.936758e+06 | 1.9367 |
| **mean** | 3.341651e+06 | 2008.0 | 6.111106e+00 | 1.575347e+01 | 3.984827e+00 | 1.518534e+03 | 1.467473e+03 | 1.610141e+03 | 1.634225e+03 | 2.1842 |
| **std** | 2.066065e+06 | 0.0 | 3.482546e+00 | 8.776272e+00 | 1.995966e+00 | 4.504853e+02 | 4.247668e+02 | 5.481781e+02 | 4.646347e+02 | 1.9447 |
| **min** | 0.000000e+00 | 2008.0 | 1.000000e+00 | 1.000000e+00 | 1.000000e+00 | 1.000000e+00 | 0.000000e+00 | 1.000000e+00 | 0.000000e+00 | 1.0000 |
| **25%** | 1.517452e+06 | 2008.0 | 3.000000e+00 | 8.000000e+00 | 2.000000e+00 | 1.203000e+03 | 1.135000e+03 | 1.316000e+03 | 1.325000e+03 | 6.1000 |
| **50%** | 3.242558e+06 | 2008.0 | 6.000000e+00 | 1.600000e+01 | 4.000000e+00 | 1.545000e+03 | 1.510000e+03 | 1.715000e+03 | 1.705000e+03 | 1.5430 |
| **75%** | 4.972467e+06 | 2008.0 | 9.000000e+00 | 2.300000e+01 | 6.000000e+00 | 1.900000e+03 | 1.815000e+03 | 2.030000e+03 | 2.014000e+03 | 3.4220 |
| **max** | 7.009727e+06 | 2008.0 | 1.200000e+01 | 3.100000e+01 | 7.000000e+00 | 2.400000e+03 | 2.359000e+03 | 2.400000e+03 | 2.400000e+03 | 2.7430 |

## Troba quantes dades faltants hi ha per columna

```
1 miss_values_count = df.isnull().sum(min_count=1)
2 miss_values_count = miss_values_count[miss_values_count != 0]
3 print(miss_values_count)
```

```
ArrTime               7110
TailNum                  5
ActualElapsedTime     8387
CRSElapsedTime         198
AirTime               8387
ArrDelay              8387
TaxiIn                7110
TaxiOut                455
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...)

```
1 # Calcul de velocitar mitjana de vol
2 df['VelocitatMitja'] = df['Distance']/df['AirTime']
3 # df['retraso']
4
5 print('primeros 5 registros de velocidad')
6 df[['VelocitatMitja', 'Distance', 'AirTime']].head()
```

primeros 5 registros de velocidad

| | VelocitatMitja | Distance | AirTime |
|---|---|---|---|
| **0** | 6.982759 | 810 | 116.0 |
| **1** | 7.168142 | 810 | 113.0 |
| **2** | 6.776316 | 515 | 76.0 |
| **3** | 6.688312 | 515 | 77.0 |
| **4** | 7.908046 | 688 | 87.0 |

## Taula de les aerolínies amb més endarreriments acumulats

```
1 dfUniqueCarrier = df.UniqueCarrier
2
```

```
print('\nRelación de compañías aéreas')
dfUniqueCarrier
```

```
Relación de compañías aéreas
0          WN
1          WN
2          WN
3          WN
4          WN
           ..
1936753    DL
1936754    DL
1936755    DL
1936756    DL
1936757    DL
```

## Quins són els vols més llargs? I els més endarrerits?

```
#Elimino las columnas que no necesito ahora

borrarColumnas = ['CancellationCode', 'Diverted', 'CarrierDelay',
       'WeatherDelay', 'NASDelay', 'SecurityDelay', 'LateAircraftDelay','CRSArrTime', 'UniqueCarrier', 'FlightNum','TaxiIn', 'TaxiOut',
                  'Unnamed: 0', 'DayofMonth', 'TailNum']

df1 = df.drop(borrarColumnas, axis=1)

#ordeno por retraso
df1=df1.sort_values(by = 'ArrDelay', ascending= False, ignore_index=True)
df1['ArrDelay']

retrasoLagos = df1.head(10)
print('\nListado de los 10 vuelos con más retraso, su origen y destino')
retrasoLagos[['ArrDelay','Origin', 'Dest' ]]
```

Listado de los 10 vuelos con más retraso, su origen y destino

| | ArrDelay | Origin | Dest |
|---|---|---|---|
| **0** | 2461.0 | HNL | MSP |

```
#Odeno por distacia
df1=df1.sort_values(by = 'Distance', ascending= False, ignore_index=True)
df1['Distance']

retrasoLagos = df1.head(10)
print('\nListado de los 10 vuelos más lasrgos en distancia, con su origen y destino')
retrasoLagos[['Distance','Origin', 'Dest' ]]
```

Listado de los 10 vuelos más lasrgos en distancia, con su origen y destino

| | Distance | Origin | Dest |
|---|---|---|---|
| **0** | 4962 | EWR | HNL |
| **1** | 4962 | EWR | HNL |
| **2** | 4962 | EWR | HNL |
| **3** | 4962 | HNL | EWR |
| **4** | 4962 | EWR | HNL |
| **5** | 4962 | EWR | HNL |
| **6** | 4962 | EWR | HNL |
| **7** | 4962 | EWR | HNL |
| **8** | 4962 | HNL | EWR |

## Dibujo la cantidad de vuelos que ha hecho cada compañía en el 2008

```
# Cuento las compañias que hay y la frecuencai con que se repite:

numero = df.groupby(df['UniqueCarrier']).count()

# Me quedo solo con la columna que necesito
borrarColumnas = ['Year', 'Month', 'DayofMonth', 'DayOfWeek', 'DepTime',
                  'CRSDepTime', 'ArrTime', 'CRSArrTime', 'FlightNum', 'TailNum',
                  'ActualElapsedTime', 'CRSElapsedTime', 'AirTime', 'ArrDelay',
                  'DepDelay', 'Origin', 'Dest', 'Distance', 'TaxiIn', 'TaxiOut',
                  'Cancelled', 'CancellationCode', 'Diverted', 'CarrierDelay',
```

```
                  'WeatherDelay', 'NASDelay', 'SecurityDelay', 'LateAircraftDelay',
                  'VelocitatMitja']

df2 = numero.drop(borrarColumnas, axis=1)

df2 = df2.rename(columns={'Unnamed: 0': 'cantidadVuelos'})


print('Cantidad de vuelos realizados por cada compañía')
print(df2)
```

```
Cantidad de vuelos realizados por cada compañía
               cantidadVuelos
UniqueCarrier
9E                      51885
AA                     191865
AQ                        750
AS                      39293
B6                      55315
CO                     100195
DL                     114238
EV                      81877
F9                      28269
FL                      71284
HA                       7490
MQ                     141920
NW                      79108
OH                      52657
OO                     132433
UA                     141426
US                      98425
WN                     377602
XE                     103663
YV                      67063
```

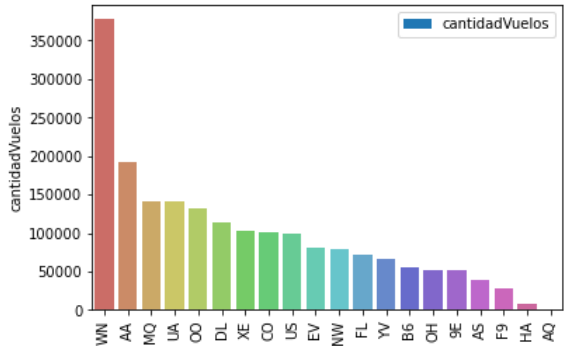
```
# Convierto el indice en una columna del df2

df2 = df2.reset_index()
df2.columns
```

```
Index(['UniqueCarrier', 'cantidadVuelos'], dtype='object')
```

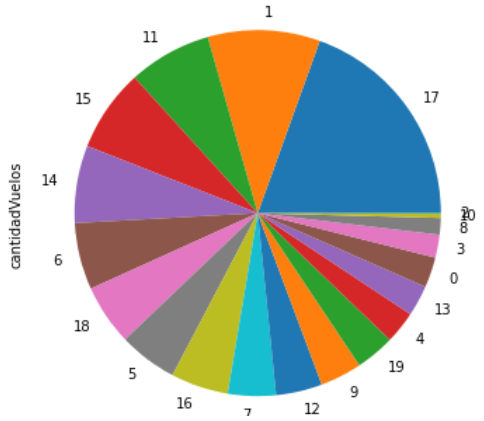
```
# ordeno y creo un grafico de barras.

df2 = df2.sort_values(by='cantidadVuelos', ascending=False)
df2['cantidadVuelos']

df2.plot(kind='bar')

```

```
8 sns.barplot(x = "UniqueCarrier", y = "cantidadVuelos", palette="hls", data = df2)
9
```

```
<AxesSubplot:xlabel='UniqueCarrier', ylabel='cantidadVuelos'>
```




```
1 plot = df2['cantidadVuelos'].plot.pie(subplots=True, figsize=(11, 6))
```




## Investigo cuantos años hay en el df

```
1 dfAnyos= df.groupby(df['Year']).count()
2 dfAnyos
```

| Year | Unnamed: 0 | Month | DayofMonth | DayOfWeek | DepTime | CRSDepTime | ArrTime | CRSArrTime | UniqueCarrier | FlightNum | ... | TaxiOut | C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Veo que solo hay un año, el 2008, con 1.936.758 vuelos

## Exercici 3

## Guardar a EXCEL¶

Exporta el data set net i amb les noves columnes a Excel.

```
# Guardar excel

df2.to_excel(r"D:\Documentos D\GitHub\0204_programaci-_Num-rica\VDatos_0205.xlsx")
print()
print('Guardado fichero : Vueling_Python.xlsx')
print()
```

```
Guardado fichero : Vueling_Python.xlsx
```

```
1
```