airtrub-1

November 27, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

1 1. Analizar los archivos y crear un dataframe para cada uno de los archivos.

[3]:
```
df_fecha = pd.read_excel(
    'C:/Users/monts/Downloads/airTribu.xlsx',
    sheet_name='airTribu',
    usecols='A:B',
    header=2
)
df_fecha.reset_index(drop=True, inplace=True)
df_fecha
```

[3]:

	vuelo	día
0	6005	2
1	2021	25
2	5528	18
3	7965	25
4	8717	13
..	…	…
95	5401	20
96	7050	24
97	1448	22
98	6970	22
99	9531	4

[100 rows x 2 columns]

[4]:
```
df_paises = pd.read_excel(
    'C:/Users/monts/Downloads/airTribu.xlsx',
    sheet_name='airTribu',
    usecols='D:E',
```

```
    header= 2
)

df_paises
```

[4]:
```
   COD_país       país
0       ARG  Argentina
1       AUS  Australia
2       BRA     Brasil
3       CAN     Canada
4       CHI      Chile
..      …         …
95      NaN        NaN
96      NaN        NaN
97      NaN        NaN
98      NaN        NaN
99      NaN        NaN

[100 rows x 2 columns]
```

[5]:
```
df_vuelos = pd.read_excel(
    'C:/Users/monts/Downloads/airTribu.xlsx',
    sheet_name='airTribu',
    usecols='G:I',
    header= 2
)

df_vuelos
```

[5]:
```
    vuelo.1 origen  destino
0      6005     MEX      ESP
1      2021     COL      MEX
2      5528     ARG      COL
3      7965     BRA      ARG
4      8717     USA      BRA
..      …      …        …
95     5401     CAN      USA
96     7050     CHI      VEN
97     1448     CHN      USA
98     6970     COL      IND
99     9531     ESP      MEX

[100 rows x 3 columns]
```

[6]:
```
df_retrasos = pd.read_excel(
    'C:/Users/monts/Downloads/airTribu.xlsx',
    sheet_name='airTribu',
```

```
    usecols='K:L',
    header= 2
)

df_retrasos
```

[6]:
```
    vuelo.2  retraso
0      6005       52
1      2021       27
2      5528       36
3      7965       82
4      8717       29
..      …       …
95     5401       58
96     7050       68
97     1448       53
98     6970       79
99     9531       83

[100 rows x 2 columns]
```

2 2. Análisis Básico con Spark, necesitamos responder a cuatro preguntas clave sobre las operaciones :

2.1 ¿Cuál fue el pais que registro el mayor número de despegues?

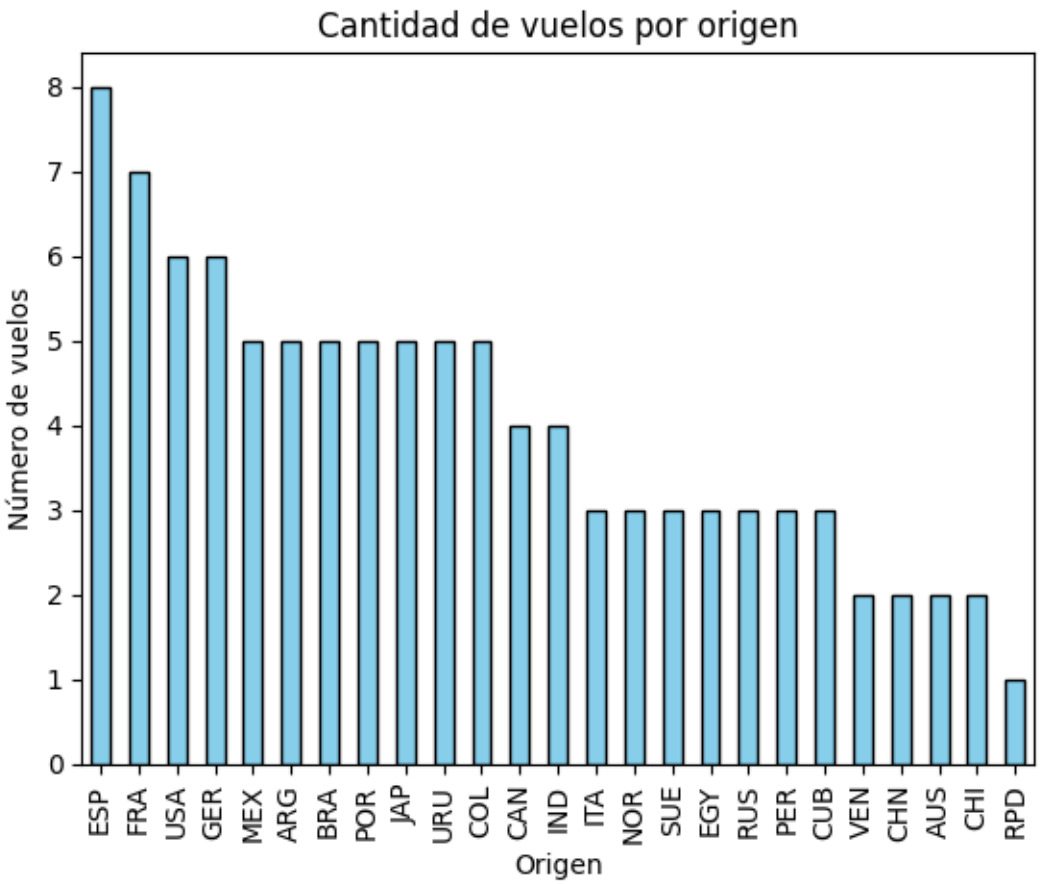
[9]:
```
df_vuelos_copia = df_vuelos[['vuelo.1', 'origen ']].copy()

conteo_vuelos = df_vuelos_copia['origen '].value_counts()

conteo_vuelos.plot(kind='bar', color='skyblue', edgecolor='black')

plt.title('Cantidad de vuelos por origen')
plt.xlabel('Origen')
plt.ylabel('Número de vuelos')

plt.show()
print(conteo_vuelos)
```




```
origen
ESP    8
FRA    7
USA    6
GER    6
MEX    5
ARG    5
BRA    5
POR    5
JAP    5
URU    5
COL    5
CAN    4
IND    4
ITA    3
NOR    3
SUE    3
EGY    3
RUS    3
```

```
PER    3
CUB    3
VEN    2
CHN    2
AUS    2
CHI    2
RPD    1
Name: count, dtype: int64
```

2.2 El país que registro mayor cantidad de despegues fue españa con 8 vuelos

3 ¿Cuál fue el país que recibió la mayor cantidad de aterrizajes?

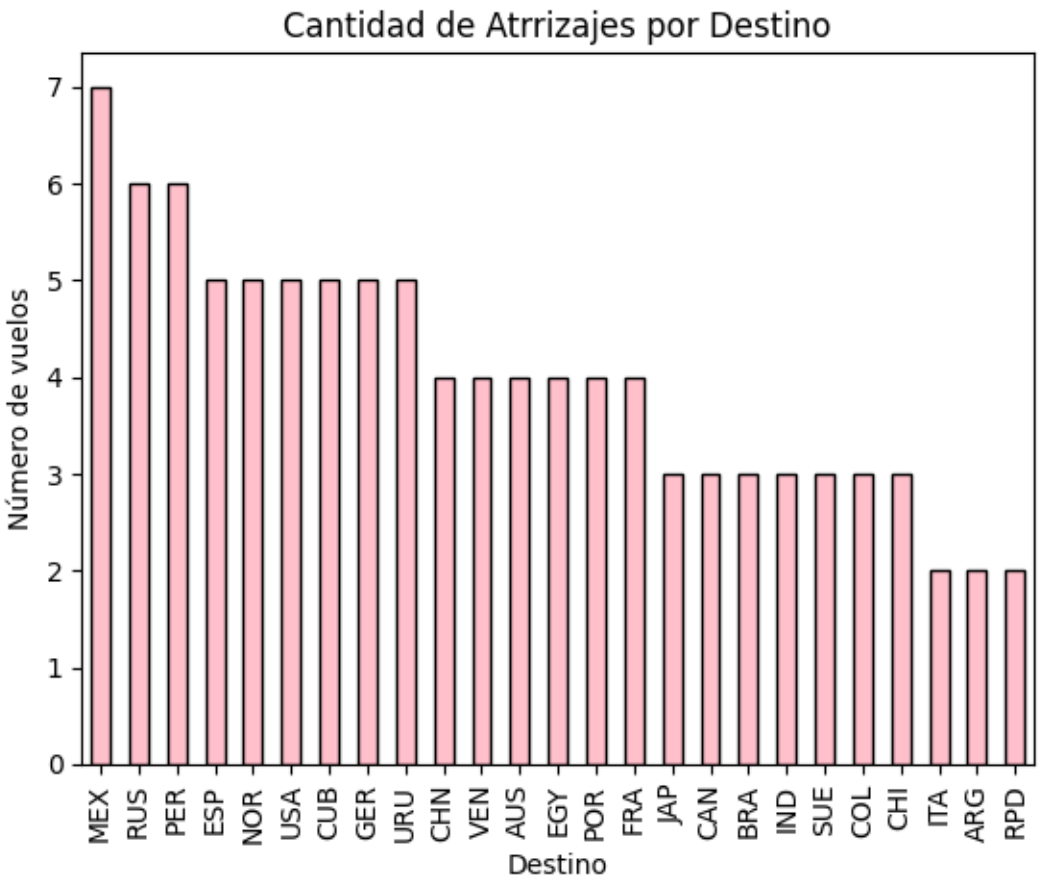
[12]:
```
df_aterrizajes_copia = df_vuelos[['origen ','destino']].copy()

conteo_aterrizajes= df_aterrizajes_copia['destino'].value_counts()

conteo_aterrizajes.plot(kind='bar', color='pink', edgecolor='black')

plt.title('Cantidad de Atrrizajes por Destino')
plt.xlabel('Destino')
plt.ylabel('Número de vuelos')

plt.show()
print(conteo_aterrizajes)
```




```
destino
MEX    7
RUS    6
PER    6
ESP    5
NOR    5
USA    5
CUB    5
GER    5
URU    5
CHN    4
VEN    4
AUS    4
EGY    4
POR    4
FRA    4
JAP    3
CAN    3
BRA    3
```

```
IND    3
SUE    3
COL    3
CHI    3
ITA    2
ARG    2
RPD    2
Name: count, dtype: int64
```

3.1 El país que registro mayor cantidad de aterrizajes fue México con 7 descensos

4 Identificar el día del mes con mayor actividad de vuelos y el día con menor actividad

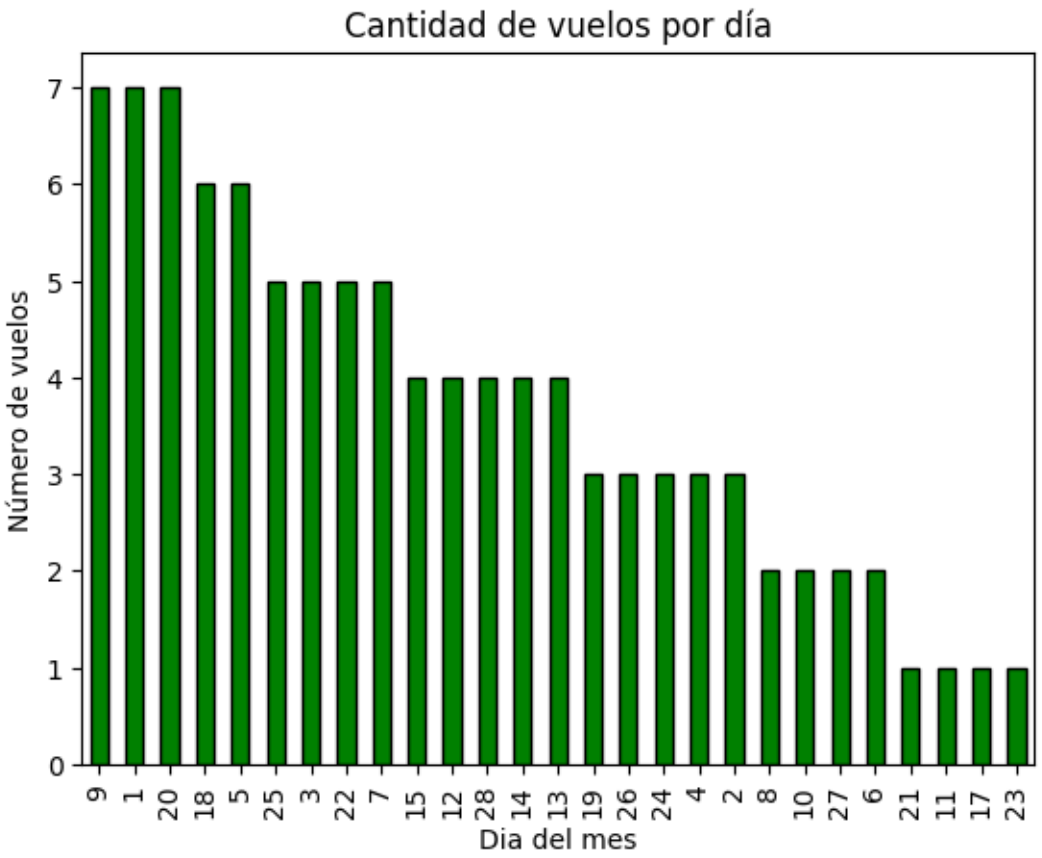
[15]:
```
df_fecha_copia = df_fecha[['vuelo', 'día']].copy()

conteo_fechas = df_fecha_copia['día'].value_counts()

conteo_fechas.plot(kind='bar', color='green', edgecolor='black')

plt.title('Cantidad de vuelos por día')
plt.xlabel('Dia del mes')
plt.ylabel('Número de vuelos')

plt.show()
print(conteo_fechas)
```



día	
9	7
1	7
20	7
18	6
5	6
25	5
3	5
22	5
7	5
15	4
12	4
28	4
14	4
13	4
19	3
26	3
24	3
4	3
2	3

```
8     2
10    2
27    2
6     2
21    1
11    1
17    1
23    1
Name: count, dtype: int64
```

4.1 Los días que registraron más vuelso en el mes de febrero del 2019, fueron el día 1, 9 y 20. Mientras que los días 11,17,23 y 21 solo tuvieron 1 vuelo al día

5 Identificar el día del mes con mayor cantidad de retrasos y cual el menor

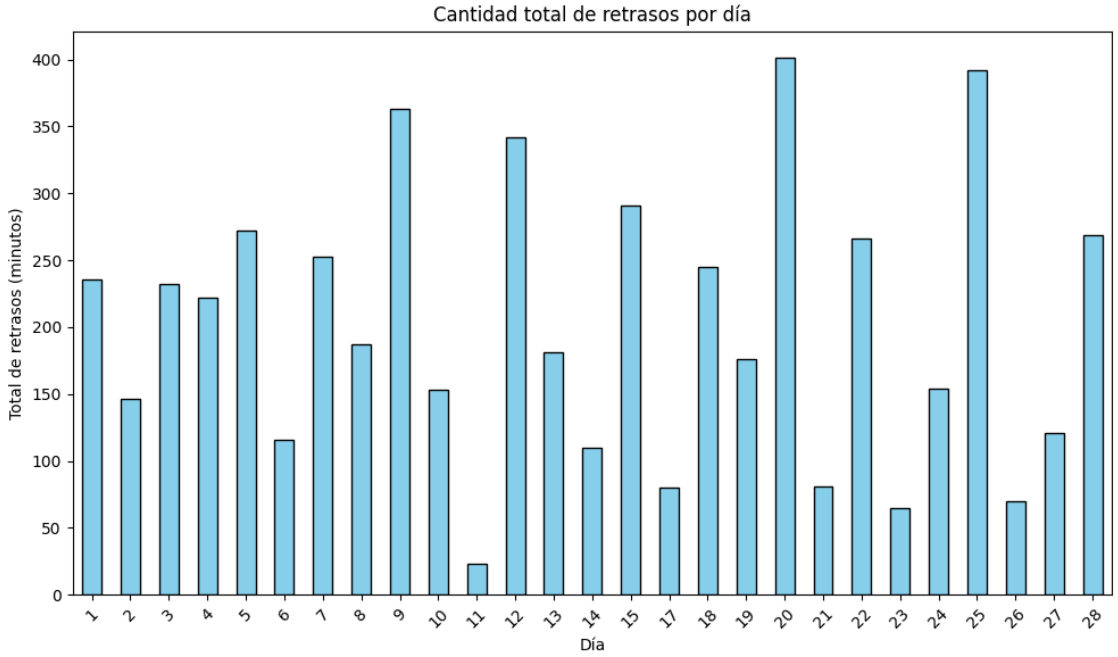
[23]:
```
df_retrasos.rename(columns={'vuelo.2': 'vuelo'}, inplace=True)
df_retrasos_copia = df_retrasos[['vuelo', 'retraso']].copy()

df_combinado = pd.merge(df_fecha_copia, df_retrasos_copia, on='vuelo',␣
 ↪how='inner')
retrasos_por_dia = df_combinado.groupby('día')['retraso'].sum()

plt.figure(figsize=(10, 6))
retrasos_por_dia.plot(kind='bar', color='skyblue', edgecolor='black')
plt.title('Cantidad total de retrasos por día')
plt.xlabel('Día')
plt.ylabel('Total de retrasos (minutos)')

plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

print(retrasos_por_dia)
```



día	
1	236
2	146
3	232
4	222
5	272
6	116
7	253
8	187
9	363
10	153
11	23
12	342
13	181
14	110
15	291
17	80
18	245
19	176
20	401
21	81
22	266
23	65
24	154
25	392
26	70

```
27    121
28    269
Name: retraso, dtype: int64
```

5.1 El día con mayor tiempo de retraso fue el día 20 con 401, mientras que el día con menor retraso fue el 11 con 23

6 Se requiere crear un nuevo DataFrame que se centre en el análisis de retrasos.

[63]:
```
df_vuelos.rename(columns={'vuelo.1': 'vuelo'}, inplace=True)
df_vuelos.rename(columns={'origen ': 'origen'}, inplace=True)
df_fecha.rename(columns={'día': 'dia'}, inplace=True)

df_totalret = df_vuelos.merge(df_retrasos, how = 'inner', on = 'vuelo').
 ↪merge(df_fecha, how = 'inner', on= 'vuelo')


df_totalret = df_totalret[["origen", "dia","retraso"]]
df_totalret.columns = ["origen", "dia","retraso"]

df_totalret = df_totalret.sort_values(by= ["origen","dia"])

df_totalret["retraso acumulado"] = df_totalret.groupby("origen")["retraso"].
 ↪cumsum()

# Configuración para mostrar todas las filas
pd.set_option('display.max_rows', None)

df_totalret
```

[63]:

	origen	dia	retraso	retraso acumulado
13	ARG	9	71	71
20	ARG	17	80	151
2	ARG	18	36	187
98	ARG	19	90	277
35	ARG	25	60	337
48	AUS	21	81	81
99	AUS	26	19	100
14	BRA	7	51	51
21	BRA	9	74	125
100	BRA	9	48	173
3	BRA	25	82	255
36	BRA	25	48	303
40	CAN	5	42	42
7	CAN	13	19	61
25	CAN	20	78	139

27	CAN	20	97	236
101	CAN	20	58	294
26	CAN	25	78	372
28	CAN	25	97	469
102	CHI	24	68	68
56	CHI	28	37	105
52	CHN	8	97	97
103	CHN	22	53	150
12	COL	7	91	91
19	COL	14	25	116
104	COL	22	79	195
34	COL	24	20	215
1	COL	25	27	242
46	CUB	1	24	24
79	CUB	3	97	121
90	CUB	20	15	136
91	EGY	1	22	22
45	EGY	19	68	90
80	EGY	22	35	125
81	ESP	1	27	27
105	ESP	4	83	110
89	ESP	5	28	138
58	ESP	7	50	188
30	ESP	9	44	232
42	ESP	12	64	296
92	ESP	13	73	369
9	ESP	28	87	456
82	FRA	1	62	62
38	FRA	3	19	81
93	FRA	5	71	152
16	FRA	9	24	176
59	FRA	14	14	190
23	FRA	15	98	288
5	FRA	26	35	323
94	GER	3	66	66
60	GER	12	99	165
83	GER	20	36	201
10	GER	22	65	266
43	GER	26	16	282
31	GER	28	50	332
95	IND	6	18	18
61	IND	15	19	37
53	IND	18	20	57
84	IND	22	34	91
96	ITA	1	33	33
51	ITA	10	96	129
62	ITA	23	65	194

63	JAP	2	20	20
41	JAP	5	37	57
29	JAP	7	16	73
97	JAP	12	95	168
8	JAP	14	50	218
0	MEX	2	52	52
65	MEX	10	57	109
64	MEX	12	84	193
18	MEX	14	21	214
33	MEX	28	95	309
68	NOR	1	36	36
55	NOR	20	50	86
66	NOR	20	29	115
47	PER	1	32	32
67	PER	13	60	92
69	PER	24	66	158
39	POR	4	67	67
17	POR	18	65	132
24	POR	18	14	146
6	POR	19	18	164
70	POR	20	78	242
72	POR	20	97	339
71	POR	25	78	417
73	POR	25	97	514
57	RPD	4	72	72
74	RUS	2	74	74
85	RUS	7	45	119
49	RUS	15	89	208
75	SUE	8	90	90
54	SUE	9	52	142
86	SUE	9	50	192
76	URU	3	37	37
87	URU	5	12	49
32	URU	15	85	134
11	URU	18	28	162
44	URU	18	82	244
15	USA	3	13	13
77	USA	6	98	111
37	USA	11	23	134
4	USA	13	29	163
22	USA	27	74	237
88	USA	27	47	284
50	VEN	5	82	82
78	VEN	20	38	120

7 4. Crear un dataframe con los valores de origen, destino y país, añade otra columna que sea "Pais_VIP"

[86]:
```
VIP = ['ESP','PER','MEX']
df_vip = df_vuelos.copy()

df_vip["VIP"]= df_vip.apply(
    lambda row : "VIP" if row["origen"] in VIP or row["destino"] in VIP else␣
 ↪'---', axis = 1
)

df_vip
```

[86]:

	vuelo	origen	destino	VIP
0	6005	MEX	ESP	VIP
1	2021	COL	MEX	VIP
2	5528	ARG	COL	---
3	7965	BRA	ARG	---
4	8717	USA	BRA	---
5	6549	FRA	USA	---
6	7566	POR	FRA	---
7	2886	CAN	POR	---
8	3122	JAP	CAN	---
9	2947	ESP	JAP	VIP
10	4380	GER	ESP	VIP
11	7869	URU	GER	---
12	8440	COL	URU	---
13	8237	ARG	GER	---
14	2415	BRA	URU	---
15	4001	USA	EGY	---
16	6645	FRA	CUB	---
17	2979	POR	PER	VIP
18	6937	MEX	AUS	VIP
19	2644	COL	RUS	---
20	1924	ARG	VEN	---
21	8716	BRA	PER	VIP
22	3928	USA	AUS	---
23	6523	FRA	RUS	---
24	3334	POR	VEN	---
25	5306	CAN	ITA	---
26	2235	JAP	CHN	---
27	1050	ESP	IND	VIP
28	5584	GER	SUE	---
29	9712	URU	NOR	---
30	1190	MEX	CHI	VIP
31	1338	COL	RPD	---
32	8559	ARG	NOR	---

33	7439	BRA	ARG	---
34	4958	USA	BRA	---
35	5726	FRA	USA	---
36	8156	POR	FRA	---
37	4329	CAN	POR	---
38	8595	JAP	CAN	---
39	5136	ESP	JAP	VIP
40	1443	GER	ESP	VIP
41	4080	URU	GER	---
42	6775	EGY	URU	---
43	3825	CUB	EGY	---
44	1148	PER	CUB	VIP
45	7871	AUS	PER	VIP
46	3529	RUS	AUS	---
47	3197	VEN	RUS	---
48	2099	ITA	NOR	---
49	7472	CHN	PER	VIP
50	5927	IND	POR	---
51	4705	SUE	RPD	---
52	6511	NOR	RUS	---
53	2157	CHI	SUE	---
54	3729	RPD	URU	---
55	6230	ESP	USA	VIP
56	9026	FRA	VEN	---
57	2429	GER	AUS	---
58	8198	IND	BRA	---
59	9556	ITA	CAN	---
60	7653	JAP	CHI	---
61	2777	MEX	CHN	VIP
62	6234	MEX	COL	VIP
63	6821	NOR	CUB	---
64	9636	PER	EGY	VIP
65	5712	NOR	ESP	VIP
66	4390	PER	FRA	VIP
67	5306	POR	GER	---
68	5165	RUS	CHI	---
69	8680	SUE	CHN	---
70	5749	URU	COL	---
71	2328	USA	CUB	---
72	6815	VEN	EGY	---
73	6194	CUB	ESP	VIP
74	9992	EGY	FRA	---
75	7824	ESP	GER	VIP
76	8432	FRA	IND	---
77	3566	GER	ITA	---
78	5138	IND	JAP	---
79	2617	RUS	MEX	VIP

```
80   6213    SUE     NOR  ---
81   1769    URU     PER  VIP
82   5600    USA     POR  ---
83   6038    ESP     MEX  VIP
84   4086    CUB     MEX  VIP
85   3521    EGY     CHN  ---
86   6651    ESP     MEX  VIP
87   7608    FRA     RUS  ---
88   6075    GER     CUB  ---
89   9962    IND     MEX  VIP
90   1713    ITA     NOR  ---
91   6554    JAP     PER  VIP
92   4861    ARG     RUS  ---
93   4481    AUS     SUE  ---
94   5039    BRA     URU  ---
95   5401    CAN     USA  ---
96   7050    CHI     VEN  ---
97   1448    CHN     USA  ---
98   6970    COL     IND  ---
99   9531    ESP     MEX  VIP
```

8 5. Si se desea almacenar la información del resultado del ejercicio 4 en solo 1 archivo, ¿cómo lo harías?

¿Y si ahora lo quisiera en 10?

[90]:
```
from IPython.display import FileLink

df_vip.to_excel('df_vip.xlsx', index=False)
FileLink(r'df_vip.xlsx')
```

[90]: C:\Users\monts\df_vip.xlsx

[95]:
```
for i in range(10):
    file_name = f"df_vip{i}.xlsx"
    df_vip.to_excel(file_name, index=False)
    display(FileLink(file_name))
```

C:\Users\monts\df_vip0.xlsx

C:\Users\monts\df_vip1.xlsx

C:\Users\monts\df_vip2.xlsx

C:\Users\monts\df_vip3.xlsx

C:\Users\monts\df_vip4.xlsx

C:\Users\monts\df_vip5.xlsx

```
C:\Users\monts\df_vip6.xlsx

C:\Users\monts\df_vip7.xlsx

C:\Users\monts\df_vip8.xlsx

C:\Users\monts\df_vip9.xlsx
```

9 6. Se requiere un proceso específico de filtrado y almacenamiento

[102]:
```
FIL = ['MEX', 'PER']
df_fil = df_vuelos.copy()

df_fil["FIL"] = df_fil.apply(
    lambda row : "FILL" if row["origen"] in FIL else "Null", axis = 1
)

df_fil = df_fil[df_fil["FIL"]== "FIL"]

df_fil
```

[102]:
```
Empty DataFrame
Columns: [vuelo, origen, destino, FIL]
Index: []
```

[98]:
```
df_vuelos.columns
```

[98]:
```
Index(['vuelo', 'origen', 'destino'], dtype='object')
```

[ ]:
```

```