Capstone I

Análisis de datos de Fórmula 1

Luis de la Ossa
Máster en Ciencia de Datos e Ingeniería de Datos en la Nube
Universidad de Castilla-La Mancha

Marta Bellón Castro
Curso 2022-2023

Índice

In [1]:
```
from IPython.display import display, HTML
display(HTML("<style>.container {width:95%} </style>"))
%config InlineBackend.figure_format = 'retina'
%matplotlib inline

import pandas as pd
import numpy as np
```

```
import matplotlib.pyplot as plt
from matplotlib import style
style.use('seaborn-whitegrid')
```

---

1. Introducción

En este primer proyecto del máster se trabajará con información sobre la competición de *Fórmula 1* generada desde el año 1950 hasta esta misma temporada. El conjunto datos se ha descargado de Kaggle (enlace), aunque a su vez el autor los recopiló del sitio `ergast.com`, que es una base de datos sobre Fórmula 1 que proporciona una API para la consulta (enlace).

A partir de este conjunto de datos, compuesto por 14 archivos `.csv`, se dará respuesta a una serie de cuestiones generales. También, como objetivo más concreto, se proporcionará una visión sobre la trayectoria (hasta el momento) del piloto español Fernando Alonso en esta competición.

El enfoque de este trabajo, al igual que del primer módulo en su totalidad, es principalmente ***instrumental***. Por tanto, en lugar de llevar a cabo un análisis exploratorio de los datos como tal (esto lo haremos en el Capstone III), se abordarán una serie de ejercicios cuya resolución implicará manipular distintos *DataFrames*, y permitirá poner en práctica las habilidades adquiridas con *pandas* y *matplotlib*. En concreto, se pondrá el foco en *pandas*, ya que la visualización se trabajará en los módulos correspondientes.

Por último, y debido a que éste es el primero de varios proyectos, **la carga de trabajo es importante**. No obstante, os animamos a ampliar el trabajo y construir alguna gráfica más en relación a algún aspecto que despierte vuestra curiosidad. Además, puesto que **una posible linea de TFM es la elaboración de un portfolio** con ampliación de los trabajos de los distintos módulos, podéis enfocar este trabajo adicional en esa línea.

El texto marcado como corresponde a un desplegable con ***sugerencias*** que pueden ayudar a resolver los ejercicios. Os animamos, no obstante, a hacer un intento de resolverlos antes de leerlas.

---

2. Conductores

El archivo `data/drivers.csv` contiene información identificativa básica sobre los pilotos. Debido a que el proyecto se centrará también en un piloto concreto, `Fernando Alonso`, se tratará esta información en primer lugar.

Ejercicio 1

Cargar los datos del archivo `data/drivers.csv` en un *DataFrame* denominado `df_drivers`, mostrar la cabecera y, posteriormente, la información sobre la estructura del *DataFrame*.

▶

In [2]:
```
# Cargo los datos del archivo
df_drivers = pd.read_csv('./data/drivers.csv', index_col=0)

# Muestro la cabecera
df_drivers.head()
```

Out[2]:

driverId	driverRef	number	code	forename	surname	dob	nationality	url
1	hamilton	44	HAM	Lewis	Hamilton	1985-01-07	British	http://en.wikipedia.org/wiki/Lewis_Hamilton
2	heidfeld	\N	HEI	Nick	Heidfeld	1977-05-10	German	http://en.wikipedia.org/wiki/Nick_Heidfeld
3	rosberg	6	ROS	Nico	Rosberg	1985-06-27	German	http://en.wikipedia.org/wiki/Nico_Rosberg
4	alonso	14	ALO	Fernando	Alonso	1981-07-29	Spanish	http://en.wikipedia.org/wiki/Fernando_Alonso
5	kovalainen	\N	KOV	Heikki	Kovalainen	1981-10-19	Finnish	http://en.wikipedia.org/wiki/Heikki_Kovalainen

In [3]:
```
# Muestro la información de la estructura
df_drivers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 854 entries, 1 to 855
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   driverRef    854 non-null    object
 1   number       854 non-null    object
 2   code         854 non-null    object
 3   forename     854 non-null    object
 4   surname      854 non-null    object
 5   dob          854 non-null    object
 6   nationality  854 non-null    object
 7   url          854 non-null    object
dtypes: object(8)
memory usage: 60.0+ KB
```

En la información del *DataFrame* puede apreciarse que una columna, `driverId`, corresponde al identificador y se ha codificado como entero. El resto, se han codificado como objetos, incluyendo el número (`number`) o la fecha de nacimiento (`dob`). Además, se aprecia que la nacionalidad (`nationality`) podría ser tratada como una categoría. Aunque todos estos cambios se podrían hacer mediante operaciones (`DataFrame.set_index()` y `DataFrame.astype()`), parece más cómodo volver a leer el archivo.

Volver a cargar el archivo `data/drivers.csv` en un *DataFrame* denominado `df_drivers`. Utilizar ahora como índice la primera columna (`driverId`), que contiene el identificador individual de cada piloto. Especificar que los valores perdidos se representan como `'\N'`, y que el tipo de datos de las columnas `driverRef`, `code`, `forename`, y `surname` debe ser `string`. Especificar también que `nationality` es una categoría, e indicar que se trate la columna `dob` como un dato temporal mediante el parámetro `parse_dates` de `pd.read_csv()`.

▶

In [4]:
```
new_types={'driverRef': 'string',
           'code': 'string',
           'forename': 'string',
           'surname': 'string',
           'nationality': 'category'}

df_drivers = pd.read_csv('./data/drivers.csv', parse_dates=['dob'], dtype=new_types, index_col='driverId', na_values='\\N')
df_drivers.head()
```

Out[4]:

driverId	driverRef	number	code	forename	surname	dob	nationality	url
1	hamilton	44.0	HAM	Lewis	Hamilton	1985-01-07	British	http://en.wikipedia.org/wiki/Lewis_Hamilton
2	heidfeld	NaN	HEI	Nick	Heidfeld	1977-05-10	German	http://en.wikipedia.org/wiki/Nick_Heidfeld
3	rosberg	6.0	ROS	Nico	Rosberg	1985-06-27	German	http://en.wikipedia.org/wiki/Nico_Rosberg
4	alonso	14.0	ALO	Fernando	Alonso	1981-07-29	Spanish	http://en.wikipedia.org/wiki/Fernando_Alonso
5	kovalainen	NaN	KOV	Heikki	Kovalainen	1981-10-19	Finnish	http://en.wikipedia.org/wiki/Heikki_Kovalainen

In [5]:
```
df_drivers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 854 entries, 1 to 855
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   driverRef    854 non-null    string
 1   number       51 non-null     float64
 2   code         97 non-null     string
 3   forename     854 non-null    string
 4   surname      854 non-null    string
 5   dob          854 non-null    datetime64[ns]
 6   nationality  854 non-null    category
 7   url          854 non-null    object
dtypes: category(1), datetime64[ns](1), float64(1), object(1), string(4)
memory usage: 55.6+ KB
```

Extraer los datos relativos al piloto `Fernando Alonso`, cuyo código es `ALO`, y almacenarlos en una *Series* denominada `alonso_data`. Almacenar el identificador correspondiente al piloto en la variable `alonso_id`.

▶

In [6]:
```
# Datos de Fernándo Alonso:
alonso_data = df_drivers[df_drivers['code'] == 'ALO']
#print(alonso_data)

# El número de Alonso
```

```
alonso_id = alonso_data.index.values[0]
print(f'El numero {alonso_id} es el identificador de Fernándo Alonso.')
```

```
El numero 4 es el identificador de Fernándo Alonso.
```

Ejercicio 2

Mostrar las filas de `df_drivers` correspondientes a los cinco pilotos más jóvenes.

▶

In [7]:
```
df_drivers['dob'].nlargest(5)
```

Out[7]:
```
driverId
852   2000-05-11
846   1999-11-13
855   1999-05-30
854   1999-03-22
853   1999-03-02
Name: dob, dtype: datetime64[ns]
```

Elaborar, utilizando *matplotlib*, un gráfico de barras que represente el número de pilotos de cada nacionalidad que hay en el *DataFrame* `df_drivers`. Utilizar solamente los datos de las 20 naciones para las que hay más pilotos, añadir el título a la figura, y configurar las etiquetas de la escala del eje X (*ticks*) para que sean legibles.

▶

In [8]:
```
df_drivers_nationality = df_drivers['nationality'].value_counts().head(20)
#print(df_drivers_nationality)

# Creo la gráfica de barras
fig, ax = plt.subplots(1,1, figsize=(10,6))

x = df_drivers_nationality.index
y = df_drivers_nationality.values

ax.bar(x,y, width=0.751, alpha=0.8, color='orange', edgecolor='black', linewidth=0.75)
ax.set_title('Pilotos por Nacionalidad', fontsize=20, pad=20);

ax.set_xlabel('Nacionalidades', fontsize=15, labelpad=10)
ax.set_ylabel('Nº de pilotos', fontsize=15, labelpad=10)
ax.tick_params(axis = 'x', direction='in', labelrotation=90, labelsize=10)


def add_value_label(x,y):
    for i in range(1, len(x)+1):
        plt.text(i-1, y[i-1]+1.5, y[i-1], ha="center", va='bottom')
```

```
add_value_label(x,y)
```

---

3. Carreras

Una vez obtenidos los datos relativos a pilotos, se trabajará con la fuente de información más importante en este conjunto de datos: los resultados de las carreras, que están recogidos en el archivo `data/results.csv`. Cada fila contiene información sobre la participación de un piloto (`driverId`) en una carrera (`raceId`). Además de la posición, contiene información relativa a tiempos, vuelta más rápida, mayor velocidad, etc, que también permitirá hacer otros análisis posteriormente.

Ejercicio 3

El archivo `data/results.csv` contiene los resultados de todas las carreras. Cargar los datos en el *DataFrame* `df_results`. Utilizar la primera columna (`resultId`) como índice. Especificar, igual que anteriormente, que los valores perdidos se representan con `'\N'`. Mostrar la cabecera.

▶

In [9]:
```
df_results = pd.read_csv('./data/results.csv', index_col='resultId', na_values='\\N')
df_results.head()
```

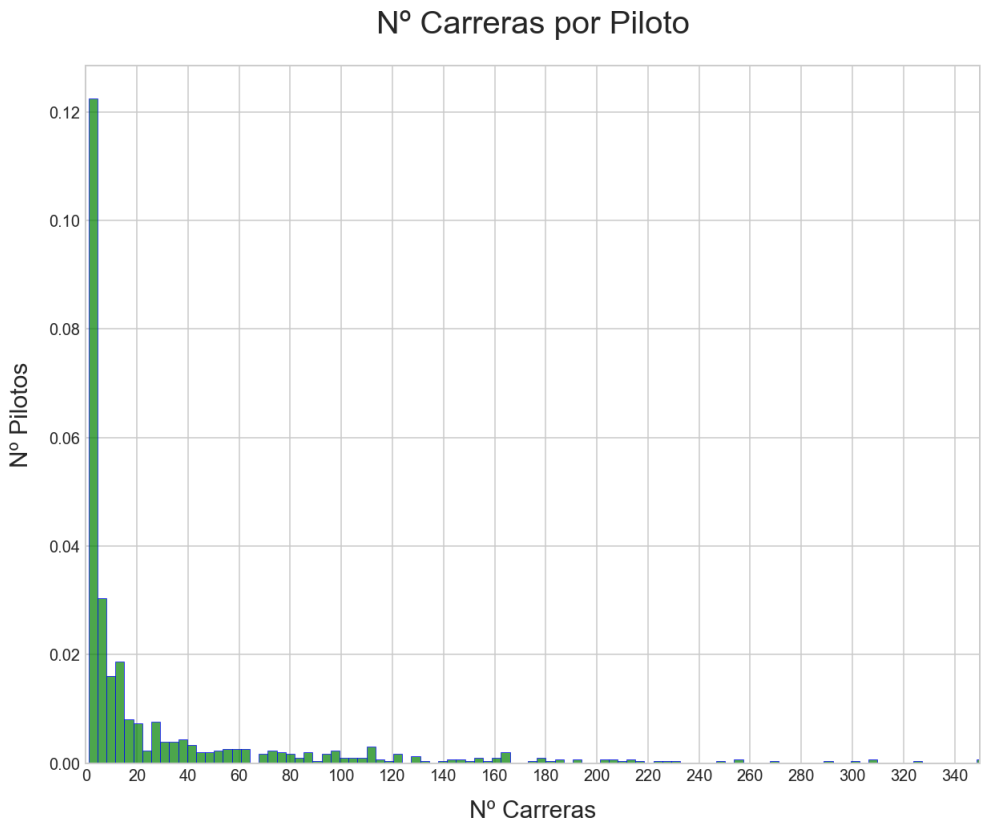
Out[9]:

resultId	raceId	driverId	constructorId	number	grid	position	positionText	positionOrder	points	laps	time	milliseconds	fastestLap	rank	fastestLapTime	fastestLapSpeed	statusId
1	18	1	1	22.0	1	1.0	1	1	10.0	58	1:34:50.616	5690616.0	39.0	2.0	1:27.452	218.300	1
2	18	2	2	3.0	5	2.0	2	2	8.0	58	+5.478	5696094.0	41.0	3.0	1:27.739	217.586	1
3	18	3	3	7.0	7	3.0	3	3	6.0	58	+8.163	5698779.0	41.0	5.0	1:28.090	216.719	1
4	18	4	4	5.0	11	4.0	4	4	5.0	58	+17.181	5707797.0	58.0	7.0	1:28.603	215.464	1
5	18	5	1	23.0	3	5.0	5	5	4.0	58	+18.014	5708630.0	43.0	1.0	1:27.418	218.385	1

Dibujar un histograma (con *matplotlib*) con el número de carreras (resultados) por piloto. Utilizar 100 intervalos. ¿Qué se aprecia?

▶

In [10]:
```
f, ax = plt.subplots(figsize=(10,8))
ax.hist(df_results[['driverId']].value_counts(), bins=100, density=True, facecolor='g', edgecolor='blue', linewidth=0.5, alpha=0.7);

# Modifico las marcas horizontales de la grafica
ax.set_xticks(range(0,350,20));

# Reduzco la grafica ya que no hay valores interesantes
ax.set_xlim(0, 350);
ax.set_title('Nº Carreras por Piloto', fontsize=20, pad=20);
ax.set_xlabel('Nº Carreras', fontsize=15, labelpad=10);
ax.set_ylabel('Nº Pilotos', fontsize=15, labelpad=10);


## Se observa como lo más frecuente es que los pilotos realicen una sola carrera. Sólo unos pocos realiza más de 300 carreras.
## La distribución es asimétrica con cola a la derecha (sesgada a la derecha)
```



Ejercicio 4

El conjunto de datos `data/races.csv` contiene información relativa a cada una de las carreras. Almacenarlo en un *DataFrame* denominado `df_races`, utilizando la columna `raceId` (la primera) como índice. Especificar que el campo `date` ha de considerarse como una fecha. Utilizar para ello el parámetro `parse_dates`. Inspeccionar el *DataFrame*.

In [11]:
```
df_races = pd.read_csv('./data/races.csv', parse_dates=['date'], index_col='raceId', na_values='\\N')
df_races.head()
```

Out[11]:

raceId	year	round	circuitId	name	date	time	url	fp1_date	fp1_time	fp2_date	fp2_time	fp3_date	fp3_time	quali_date	quali_time	sprint_date	sprint_time
1	2009	1	1	Australian Grand Prix	2009-03-29	06:00:00	http://en.wikipedia.org/wiki/2009_Australian_G...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	2009	2	2	Malaysian Grand Prix	2009-04-05	09:00:00	http://en.wikipedia.org/wiki/2009_Malaysian_Gr...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	2009	3	17	Chinese Grand Prix	2009-04-19	07:00:00	http://en.wikipedia.org/wiki/2009_Chinese_Gran...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	2009	4	3	Bahrain Grand Prix	2009-04-26	12:00:00	http://en.wikipedia.org/wiki/2009_Bahrain_Gran...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	2009	5	4	Spanish Grand Prix	2009-05-10	12:00:00	http://en.wikipedia.org/wiki/2009_Spanish_Gran...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Al inspeccionar el *DataFrame* se observa que la columna `time` está expresada como una cadena de texto. Generar una nueva *Series* con los valores de la columna `time` convertidos a marcas de tiempo. ¿Qué se aprecia?

▶

In [12]:
```
import datetime
#df_races.info()
```

In [13]:
```
# Transformo los datos a temporales, quitando la fecha de hoy con .dt.time (se escribe automaticamente con .to_datetime())
series_time= pd.to_datetime(df_races['time']).dt.time
series_time

## Se aprecia que las horas mostradas son las correspondientes al inicio de la carrera
```

Out[13]:
```
raceId
1       06:00:00
2       09:00:00
3       07:00:00
4       12:00:00
5       12:00:00
          ...
1092    05:00:00
1093    19:00:00
1094    20:00:00
1095    18:00:00
1096    13:00:00
Name: time, Length: 1079, dtype: object
```

Crear una columna denominada `df_races[date time]` en la que se represente la fecha y hora en la que tuvo lugar cada carrera. Para ello, es necesario convertir a `TimeDelta` el contenido de la columna `time`, y sumárselo a la columna `date`.

▶

In [14]:
```
df_races['date time'] = df_races['date'] + pd.to_timedelta(df_races['time'])
df_races.head()
```

Out[14]:

raceId	year	round	circuitId	name	date	time	url	fp1_date	fp1_time	fp2_date	fp2_time	fp3_date	fp3_time	quali_date	quali_time	sprint_date	sprint_time
1	2009	1	1	Australian Grand Prix	2009-03-29	06:00:00	http://en.wikipedia.org/wiki/2009_Australian_G...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	2009	2	2	Malaysian Grand Prix	2009-04-05	09:00:00	http://en.wikipedia.org/wiki/2009_Malaysian_Gr...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	2009	3	17	Chinese Grand Prix	2009-04-19	07:00:00	http://en.wikipedia.org/wiki/2009_Chinese_Gran...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	2009	4	3	Bahrain Grand Prix	2009-04-26	12:00:00	http://en.wikipedia.org/wiki/2009_Bahrain_Gran...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	2009	5	4	Spanish Grand Prix	2009-05-10	12:00:00	http://en.wikipedia.org/wiki/2009_Spanish_Gran...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Eliminar las columnas `date` y `time`. También las relativas a las horas de entrenamiento libres (`fpx_`), calificaciones (`qualy_`) y sprints (`sprint_`). Mover la nueva columna, `date time`, a la posición que ocupaba `date`. Es decir, el *DataFrame* resultante debe tener 5 columnas: `year`, `round`, `circuitId`, `date time` y `url`.

▶

In [15]:
```
df_races = df_races[['year','round','circuitId','date time','url']]
df_races.head()
```

Out[15]:

raceId	year	round	circuitId	date time	url
1	2009	1	1	2009-03-29 06:00:00	http://en.wikipedia.org/wiki/2009_Australian_G...
2	2009	2	2	2009-04-05 09:00:00	http://en.wikipedia.org/wiki/2009_Malaysian_Gr...
3	2009	3	17	2009-04-19 07:00:00	http://en.wikipedia.org/wiki/2009_Chinese_Gran...
4	2009	4	3	2009-04-26 12:00:00	http://en.wikipedia.org/wiki/2009_Bahrain_Gran...
5	2009	5	4	2009-05-10 12:00:00	http://en.wikipedia.org/wiki/2009_Spanish_Gran...

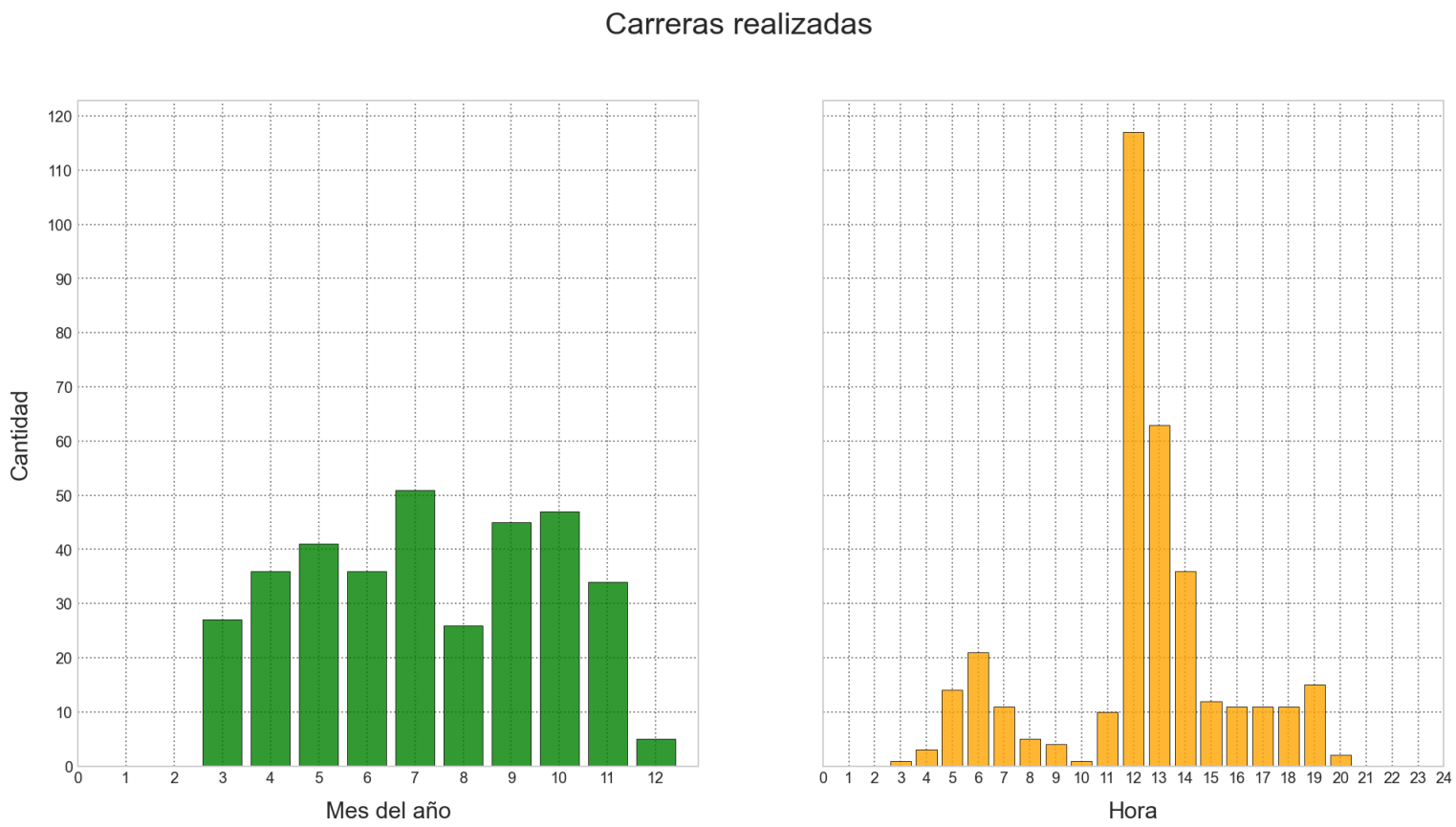
Ejercicio 5

Crear una figura con dos gráficos de barras. En el de la izquierda se han de mostrar las carreras que han tenido lugar en cada mes del año. En el de la derecha, las que han tenido lugar en cada hora. Ambas gráficas deben compartir la escala en el eje *y*.

▶

In [16]:
```
# Series de mes y hora
series_races_month = df_races['date time'].dt.month;
series_races_hour = df_races['date time'].dt.hour;

# Creo la figura con dos gráficas
fig, axes = plt.subplots(1,2, figsize=(16,8), sharey=True);

axes[0].bar(series_races_month.value_counts().index, series_races_month.value_counts().values, label='Carreras por mes', facecolor='g', edgecolor='black', linewidth=0.5, alpha=0.8
axes[1].bar(series_races_hour.value_counts().index, series_races_hour.value_counts().values, label='Carreras por hora', facecolor='orange', edgecolor='black', linewidth=0.5, alpha

fig.suptitle('Carreras realizadas', fontsize=20);
axes[0].set_xlabel('Mes del año', fontsize=15, labelpad=10);
axes[1].set_xlabel('Hora', fontsize=15, labelpad=10);
axes[0].set_ylabel('Cantidad', fontsize=15, labelpad=10);

axes[1].tick_params(axis = 'both', labelsize=10)
axes[0].set_xticks(range(0,13,1));
axes[1].set_xticks(range(0,25,1));
axes[0].set_yticks(range(0,130,10));
```

```
axes[0].grid(which='major', color='grey', linestyle=':', linewidth=1)
axes[1].grid(which='major', color='grey', linestyle=':', linewidth=1)
```



Ejercicio 6

La columna `statusId` de `df_results` contiene un código numérico que representa el estado en que cada coche/piloto concluyó cada carrera. El conjunto de datos `data/status.csv` contiene la descripción del estado. Leer este archivo y almacenarlo en el *DataFrame*. `df_status`.

In [17]:
```
df_status = pd.read_csv('./data/status.csv', index_col='statusId')
df_status.head()
```

Out[17]:

statusId	status
1	Finished
2	Disqualified
3	Accident
4	Collision
5	Engine

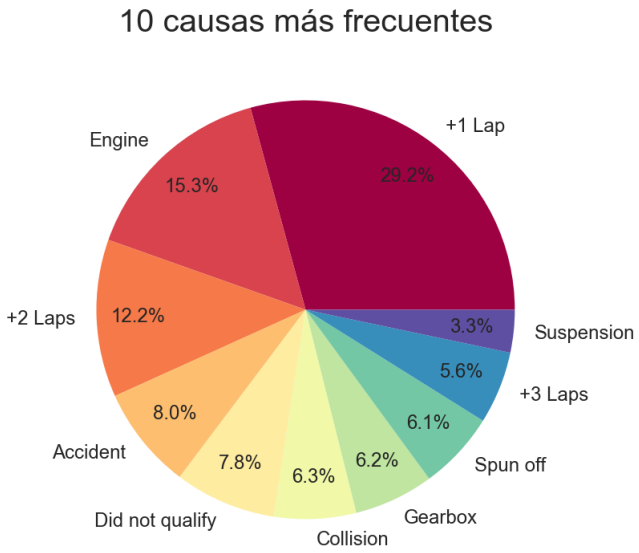
Utilizar ambos *DataFrames*, `df_results` y `df_status`, para elaborar un diagrama de sectores en el que se visualicen las 10 causas más frecuentes que han impedido terminar a los pilotos las carreras. Utilizar un tamaño de figura 6×6.

▶

In [18]:
```
# Cuento las veces que aparece el estado de terminación y obtengo los 11 mas frecuentes
df_results_finish = df_results['statusId'].value_counts()
#print(df_results_finish.head(11))

# Descarto el estado de terminación 1 y guardo el resultado como causes
causes = df_results_finish.drop(df_results_finish.index[[0]], inplace=False).head(10)
#print(causes)
```

In [19]:
```
# Obtengo los id de las causas más frecuentes
id_causes = causes.index[:].tolist()
#print(f'Los id de las causas mas frecuentes son: {id_causes}')

# Obtengo los nombres de las causas más frecuentes
status = df_status.loc[id_causes].values.tolist()
#print(f'Las causas mas frecuentes son: {status}')

# Como es una lista de listas, lo transformo en una lista
status_list = [item for sublist in status for item in sublist]
#print(f'Las causas mas frecuentes son: {status_list}')
```

In [20]:
```
import matplotlib.pyplot as plt
import numpy as np

cmap = plt.get_cmap('Spectral')
colors = [cmap(i) for i in np.linspace(0, 1, 10)]

x = causes/sum(causes)*100
y = causes

fig, ax = plt.subplots(figsize=(6,6))
ax.pie(y, shadow=False, startangle=0, autopct='%1.1f%%', pctdistance=0.8, labels=status_list, colors=colors, textprops={'fontsize': 12})
ax.set_title('10 causas más frecuentes', fontsize=20, pad=10);
```



Ejercicio 7

Mostrar el nombre, apellido, nacionalidad y número de victorias de los cinco pilotos que acumulan más victorias.

▶

In [21]:
```
#df_drivers.head()
```

In [22]:
```
#df_results.head()
```

In [23]:
```
df_results_pilots = df_results[df_results['position']==1]
#df_results_pilots.head()
```

In [24]:
```
# Obtengo los datos de id del piloto y su posición en la carrera (victoria)
df_results_pilots = df_results_pilots[['driverId', 'position']]
#df_results_pilots.head(20)
```

In [25]:
```
# Cuento las victorias de cada piloto y lo almaceno en una columna llamada wins
series = df_results_pilots['driverId'].value_counts().nlargest(5)
dataframe = series.to_frame(name= 'wins')
#dataframe
```

In [26]:
```
# Renombro el indice como driverId
dataframe.index.names = ['driverId']
```

```
#dataframe
```

In [27]:
```
# Hago el merge entre los dos df, utilizando la columna driverId como nexo de union
df_merge = pd.merge(df_drivers,dataframe, how='outer', on='driverId')

# Selecciono las columnas que me interesan
df_merge = df_merge[['forename', 'surname', 'nationality', 'wins']]

# El tipo de wins paso de entero a decimal, asi que lo cambio de nuevo a entero
df_merge['wins'] = df_merge['wins'].astype('Int64')

# Muestro la cabecera
df_merge = df_merge.sort_values(by='wins', ascending=False).head()
df_merge
```

Out[27]:

driverId	forename	surname	nationality	wins
1	Lewis	Hamilton	British	103
30	Michael	Schumacher	German	91
20	Sebastian	Vettel	German	53
117	Alain	Prost	French	51
102	Ayrton	Senna	Brazilian	41

In [28]:
```
## Resuelvo el ejercicio por method chaining

df_results_pole = pd.merge(df_drivers, df_results[df_results['position']==1][['driverId']].value_counts().nlargest(5).to_frame(name='wins'), how='inner', on='driverId')[['forename
df_results_pole.nlargest(5, 'wins')
```

Out[28]:

driverId	forename	surname	nationality	wins
1	Lewis	Hamilton	British	103
30	Michael	Schumacher	German	91
20	Sebastian	Vettel	German	53
117	Alain	Prost	French	51
102	Ayrton	Senna	Brazilian	41

Mostrar el nombre, apellido, nacionalidad y número *pole positions* de los cinco pilotos que acumulan más sesiones de calificación han ganado. La posición de un piloto en la calificación de cada carrera viene dada por la columna `grid`.

▶

In [29]:
```
df_results_pole = pd.merge(df_drivers, df_results[df_results['grid']==1][['driverId']].value_counts().nlargest(5).to_frame(name='poles'), how='inner', on='driverId')[['forename',
df_results_pole.nlargest(5, 'poles')
```

Out[29]:

driverId	forename	surname	nationality	poles
1	Lewis	Hamilton	British	103
30	Michael	Schumacher	German	68
102	Ayrton	Senna	Brazilian	65
20	Sebastian	Vettel	German	57
373	Jim	Clark	British	34

Ejercicio 8

Cuando el análisis de datos se centra en una categoría o subconjunto de datos obtenidos mediante selección condicional, puede resultar cómodo almacenar este resultado intermedio en un *DataFrame*. Además, si no se va a escribir sobre él, no es necesario hacer copia, sino que se puede hacer una asignación (Anexo I del tutorial). Almacenar los resultados correspondientes a carreras terminadas por `Fernando Alonso` en un *DataFrame* denominado `df_results_alo`. Mostrar cuántas son.

▶

In [30]:
```
#df_results.head()
```

In [31]:
```
status_id = df_status[df_status['status']=='Finished'].index.values[0]
driver_ref = df_drivers[df_drivers['driverRef']=='alonso'].index.values[0]

df_results_alo = df_results.loc[(df_results['driverId']==driver_ref) & (df_results['statusId']==status_id)]
print(f'Fernándo Alonso ha completado {len(df_results_alo)} carreras.')
```

```
Fernándo Alonso ha completado 220 carreras.
```

La columna `position` del *DataFrame* `df_results` (y de `df_results_alo`) contiene la posición en la que un conductor quedó en la carrera correspondiente. Obtener el número de podios (posiciones primera, segunda y tercera) que hizo `Fernando Alonso` a lo largo de su trayectoria deportiva (hasta el momento).

▶

In [32]:
```
df_results_alo_pos = df_results_alo.loc[(df_results_alo['position']>0) & (df_results_alo['position']<=3)]['position'].value_counts().sort_index(ascending=True)
print(f'A lo largo de su carrera Alonso obtubo: \n {df_results_alo_pos[1]} podios en primera posición. \n {df_results_alo_pos[2]} podios en segunda posición. \n {df_results_alo_p
```

```
A lo largo de su carrera Alonso obtubo:
 32 podios en primera posición.
 37 podios en segunda posición.
 29 podios en tercera posición.
```

Mostrar el número de ocasiones en las que `Fernando Alonso` ocupó las posiciones uno a cuatro en la salida (primera y segunda líneas).

▶

In [33]:
```
df_results_alo_pos = df_results_alo.loc[(df_results_alo['grid']>0) & (df_results_alo['grid']<=4)]['grid'].value_counts().sort_index(ascending=True)
print(f'A lo largo de su carrera Alonso obtubo: \n {df_results_alo_pos[1]} salidas en primera posición. \n {df_results_alo_pos[2]} salidas en segunda posición. \n {df_results_alo_
```

```
A lo largo de su carrera Alonso obtubo:
 21 salidas en primera posición.
 13 salidas en segunda posición.
 25 salidas en tercera posición.
 21 salidas en cuarta posición.
```

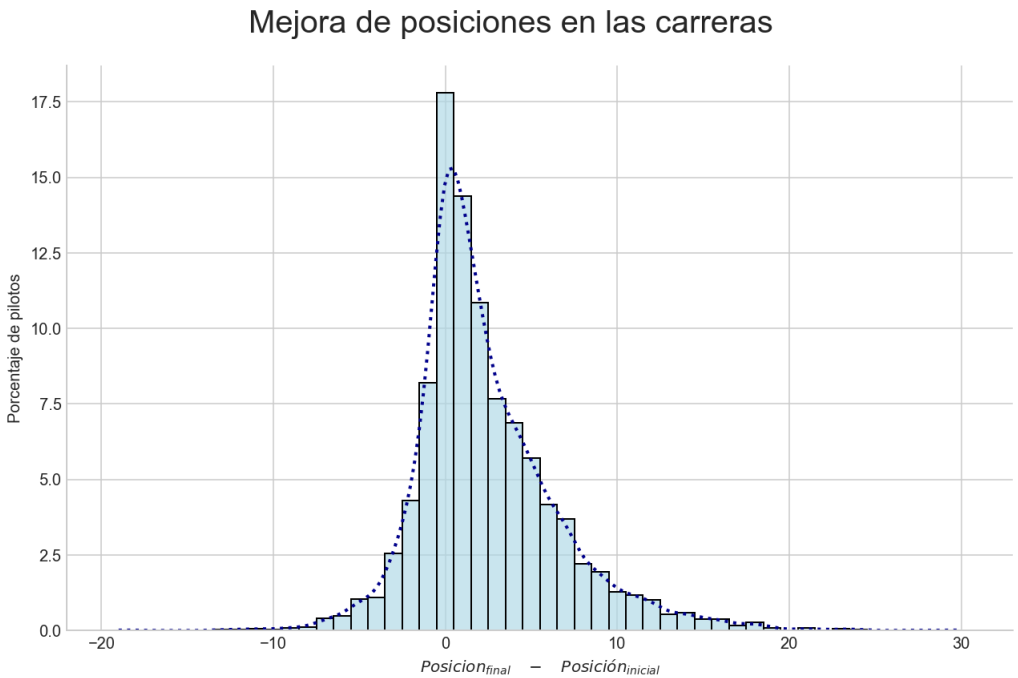
Ejercicio 9

Debido a la importancia de los coches, uno de los factores que permiten calificar la actuación de un piloto es las posiciones que gana o pierde con respecto a la posición de partida. Calcular este resultado para cada piloto/carrera, y dibujar un histograma con la distribución de valores, pero solamente para los pilotos *que acabaron cada carrera*. ¿Qué lectura podría hacerse del histograma?

▶

In [34]:
```
# Pilotos que acabaron la carrera (statusId=1)
df_results_perform=df_results.loc[df_results['statusId']==1].copy()

df_results_perform['performance'] = (df_results_perform.apply(lambda row: row.grid - row.position, axis = 1))
df_results_perform_hist = df_results_perform[['driverId','performance','statusId']]
```

In [35]:
```
import seaborn as sns
fg = sns.displot(data=df_results_perform_hist['performance'], stat="percent", discrete=True, height=6, aspect=1.5, color='lightblue', alpha=0.65,
                 kde=True, line_kws={'linewidth': 2, 'linestyle':':'})

fg.ax.lines[0].set_color('darkblue')
fg.set(xlabel='$Posicion_{final} \quad  -  \quad  Posición_{inicial}$' , ylabel = 'Porcentaje de pilotos');

fg.fig.subplots_adjust(top=.9)
fg.fig.suptitle('Mejora de posiciones en las carreras', fontsize=20);

## Se observa como la mayoría de los pilotos que finalizan las carreras lo hacen en la misma posición que la de partida, o mejoran una/dos posiciones.
```



Ejercicio 10

Calcular las posiciones que, de media, han ganado los pilotos **en las carreras finalizadas**. Mostrar el resultado para los cinco pilotos que más destacan en este aspecto, junto a su nombre, apellidos y nacionalidad.

__Nota__: Este ejercicio **es difícil**.

▶

In [36]:
```
df_results_pm=df_results.loc[(df_results['grid']<=5) & (df_results['position']<=3) & (df_results['statusId']==1)].copy()
df_results_pm['pos_won'] = df_results_pm['grid']-df_results_pm['position']
#df_results_pm
```

In [37]:
```
series_won = df_results_pm.groupby('driverId')['pos_won'].mean()
df_results_pm3 = series_won.to_frame(name= 'pos_won')
#df_results_pm3.nlargest(5, 'pos_won')
```

In [38]:
```
df_results_pm3 = pd.merge(df_results_pm3, df_drivers, how='inner', on='driverId')
#df_results_pm3
```

In [39]:
```
df_results_pw = df_results_pm3[['forename', 'surname', 'nationality', 'pos_won']]
```

In [40]:
```
df_results_pw = df_results_pw.nlargest(5, 'pos_won')#.sort_values(by='pos_won', ascending=False).nlargest(5, 'pos_won')
df_results_pw
```

Out[40]:

driverId	forename	surname	nationality	pos_won
65	Johnny	Herbert	British	4.0
593	Johnnie	Parsons	American	4.0
306	Jean-Pierre	Beltoise	French	3.0
340	John	Love	Rhodesian	3.0
723	Rudi	Fischer	Swiss	3.0

¿Cuantas carreras ha disputado y concluido el piloto más destacado en este aspecto?

In [41]:
```
driver_id = df_results_pw.index.values[0]
driver_forename = df_results_pw['forename'].values[0]
driver_surname = df_results_pw['surname'].values[0]
driver_pos_won = df_results_pw['pos_won'].values[0]
driver_id_races = df_results.loc[(df_results['driverId']==driver_id) & (df_results['statusId']==1)]['driverId'].count()

print(f'El piloto {driver_forename} {driver_surname} ha disputado {driver_id_races} carreras, y suele ganar {driver_pos_won:.2f} posiciones de media en cada carrera.')
```

```
El piloto Johnny Herbert ha disputado 29 carreras, y suele ganar 4.00 posiciones de media en cada carrera.
```

Puede apreciarse que la información está viciada porque incluye a pilotos que han competido muy pocas veces y les ha ido muy bien. Repetir el ejercicio anterior, pero utilizando solamente datos de aquellos pilotos que han finalizado al menos 75 carreras.

▶

In [42]:
```
df_results_pm = df_results.loc[(df_results['grid']<=5) & (df_results['position']<=3) & (df_results['statusId']==1)].copy()
df_results_pm['pos_won'] = df_results_pm['grid']-df_results_pm['position']
#df_results_pm
```

In [43]:
```
def check_df(df_group):
    return len(df_group)>=75

df_results_pm_75 = df_results_pm.groupby('driverId').filter(check_df)[['driverId','pos_won']].groupby('driverId')#.nlargest(5)
#df_results_pm_75
```

In [44]:
```
series_75 = df_results_pm_75['pos_won'].mean()
#series_75
```

```
df_results_75_final = series_75.to_frame(name= 'pos_won_mean')
#df_results_75_final #.nlargest(5, 'pos_won')
```

In [45]:
```
df_75 = pd.merge(df_results_75_final, df_drivers, left_on='driverId', right_on='driverId')
#df_75
```

In [46]:
```
df_75 = df_75.sort_values(by='pos_won_mean', ascending=False)#.nlargest(5, 'pos_won_mean')
#df_75
```

In [47]:
```
driver_id_75 = df_75.index.values[0]
driver_forename_75 = df_75['forename'].values[0]
driver_surname_75 = df_75['surname'].values[0]
driver_pos_won_mean = df_75['pos_won_mean'].values[0]
```

In [48]:
```
amount_races = df_results.loc[(df_results['driverId']==driver_id_75) & (df_results['statusId']==1)]['driverId'].count()
amount_races
print(f'El piloto {driver_forename_75} {driver_surname_75} ha disputado {amount_races} carreras, y suele ganar {driver_pos_won_mean:.2f} posiciones de media en cada carrera.')
```

```
El piloto Fernando Alonso ha disputado 220 carreras, y suele ganar 1.07 posiciones de media en cada carrera.
```

---

4. Escuderías

El elemento fundamental en Fórmula 1 es el coche. La diferencia entre unos equipos y otros es tan alta que, para muchos, es una competición entre escuderías.

Ejercicio 11

Leer los datos de las escudarías, almacenadas en el archivo `data/constructors.csv` y almacenarlos en un *DataFrame* denominado `df_constructors`. Utilizar la primera columna como índice.

In [49]:
```
df_constructors = pd.read_csv('./data/constructors.csv', index_col='constructorId')
#df_constructors.head()
```

Mostrar los nombres y nacionalidades de los cinco constructores que han ganado más carreras a lo largo de la historia de la competición.

▶

In [50]:
```
df_constr =  pd.merge(df_constructors, df_results[df_results['position']==1]
              .groupby('constructorId')['position']
              .agg(wins=len)
              .sort_values(by='wins', ascending=False)
              .rename(columns = {'position':'wins'}),
            left_on='constructorId', right_on='constructorId')[['name', 'nationality', 'wins']].nlargest(5, 'wins')
```

```
df_constr
```

Out[50]:

constructorId	name	nationality	wins
6	Ferrari	Italian	243
1	McLaren	British	179
131	Mercedes	German	124
3	Williams	British	114
9	Red Bull	Austrian	84

Mostrar los puntos (columna `points`) que ganó `Fernando Alonso` con cada constructor, dando además el nombre y la nacionalidad del constructor.

▶

In [51]:
```
df_results_alo_points = pd.merge(df_constructors, df_results_alo
                           .groupby(['constructorId'])['points']
                           .sum()
                           .sort_values(ascending=False)
                           .to_frame(),
                           left_on='constructorId', right_on='constructorId')
df_results_alo_points[['name', 'nationality', 'points']].nlargest(5, 'points')
```

Out[51]:

constructorId	name	nationality	points
6	Ferrari	Italian	1180.0
4	Renault	French	464.0
1	McLaren	British	205.0
214	Alpine F1 Team	French	100.0

---

5. Circuitos

Ejercicio 12

Por otra parte, el archivo `data/circuits.csv` contiene información relativa a cada uno de los circuitos. Leerlo y almacenarlo en el *DataFrame* `df_circuits`, utilizando la primera columna, `circuitId` como índice.

In [52]:
```
df_circuits = pd.read_csv('./data/circuits.csv', index_col='circuitId', na_values='\\N')
df_circuits.head()
```

Out[52]:

circuitId	circuitRef	name	location	country	lat	lng	alt	url
1	albert_park	Albert Park Grand Prix Circuit	Melbourne	Australia	-37.84970	144.96800	10.0	http://en.wikipedia.org/wiki/Melbourne_Grand_P...
2	sepang	Sepang International Circuit	Kuala Lumpur	Malaysia	2.76083	101.73800	18.0	http://en.wikipedia.org/wiki/Sepang_Internatio...
3	bahrain	Bahrain International Circuit	Sakhir	Bahrain	26.03250	50.51060	7.0	http://en.wikipedia.org/wiki/Bahrain_Internati...
4	catalunya	Circuit de Barcelona-Catalunya	Montmeló	Spain	41.57000	2.26111	109.0	http://en.wikipedia.org/wiki/Circuit_de_Barcel...
5	istanbul	Istanbul Park	Istanbul	Turkey	40.95170	29.40500	130.0	http://en.wikipedia.org/wiki/Istanbul_Park

In [53]:
```
#df_results.head()
```

In [54]:
```
#df_races.head()
```

Mostrar los 5 circuitos más rápidos, entendiendo como tales aquellos para los que la media de las 100 vueltas más rápidas (cien valores más altos de `fastestLapSpeed`) es más alta.

Nota: Obviamente, este criterio es mejorable. Pero es un ejemplo.

> __Nota__: Este ejercicio **es el más difícil**. Quizá se puede dejar para el final.

▶

In [55]:
```
df_results_races = pd.merge(df_results, df_races, how='left', on='raceId')
df_result_100 = pd.merge(df_circuits, (df_results_races[['raceId','circuitId', 'fastestLapSpeed']]
                                       .sort_values('fastestLapSpeed',ascending=False)
                                       .groupby('circuitId')['fastestLapSpeed']
                                       .apply(lambda grp: grp.nlargest(100).mean())
                                       .nlargest(5)
                                       .to_frame(name= 'mean_speed')),
                         how='right', on='circuitId')

df_result_100
```

Out[55]:

circuitId	circuitRef	name	location	country	lat	lng	alt	url	mean_speed
14	monza	Autodromo Nazionale di Monza	Monza	Italy	45.6156	9.28111	162.0	http://en.wikipedia.org/wiki/Autodromo_Naziona...	249.98647
77	jeddah	Jeddah Corniche Circuit	Jeddah	Saudi Arabia	21.6319	39.10440	15.0	http://en.wikipedia.org/wiki/Jeddah_Street_Cir...	238.65150
9	silverstone	Silverstone Circuit	Silverstone	UK	52.0786	-1.01694	153.0	http://en.wikipedia.org/wiki/Silverstone_Circuit	234.17830
13	spa	Circuit de Spa-Francorchamps	Spa	Belgium	50.4372	5.97139	401.0	http://en.wikipedia.org/wiki/Circuit_de_Spa-Fr...	233.47847
70	red_bull_ring	Red Bull Ring	Spielberg	Austria	47.2197	14.76470	678.0	http://en.wikipedia.org/wiki/Red_Bull_Ring	228.07410

Ejercicio 13

Calcular el resultado anterior (los circuitos más rápidos) con los datos relativos a `Fernando Alonso`, almacenados en `df_results_alo`.

▶

In [56]:
```
df_results_races_alo = pd.merge(df_results_alo, df_races, how='left', on='raceId')
df_result_100 = pd.merge(df_circuits, (df_results_races_alo[['raceId','circuitId', 'fastestLapSpeed']]
                                       .sort_values('fastestLapSpeed',ascending=False)
                                       .groupby('circuitId')['fastestLapSpeed']
                                       .apply(lambda grp: grp.nlargest(100).mean())
                                       .to_frame(name= 'mean_speed')
                                       .nlargest(5, 'mean_speed')),
                         how='right', on='circuitId')

df_result_100
```

Out[56]:

circuitId	circuitRef	name	location	country	lat	lng	alt	url	mean_speed
14	monza	Autodromo Nazionale di Monza	Monza	Italy	45.6156	9.28111	162.0	http://en.wikipedia.org/wiki/Autodromo_Naziona...	243.132556
9	silverstone	Silverstone Circuit	Silverstone	UK	52.0786	-1.01694	153.0	http://en.wikipedia.org/wiki/Silverstone_Circuit	226.850250
13	spa	Circuit de Spa-Francorchamps	Spa	Belgium	50.4372	5.97139	401.0	http://en.wikipedia.org/wiki/Circuit_de_Spa-Fr...	225.016143
78	losail	Losail International Circuit	Al Daayen	Qatar	25.4900	51.45420	NaN	http://en.wikipedia.org/wiki/Losail_Internatio...	223.437000
22	suzuka	Suzuka Circuit	Suzuka	Japan	34.8431	136.54100	45.0	http://en.wikipedia.org/wiki/Suzuka_Circuit	222.162714

Calcular también en qué circuitos obtuvo `Fernando Alonso`, de media, mejor posición.

▶

In [57]:
```
df_results_races_pos = pd.merge(df_results_alo, df_races, how='left', on='raceId')
df_result_pos= pd.merge(df_circuits, (df_results_races_pos[['raceId','circuitId', 'position']]
                                      .sort_values('position',ascending=False)
                                      .groupby('circuitId')['position']
                                      .mean()
                                      .to_frame(name= 'mean_pos')
                                      .nsmallest(5, 'mean_pos')),
                        how='right', on='circuitId')

df_result_pos
```

Out[57]:

circuitId	circuitRef	name	location	country	lat	lng	alt	url	mean_pos
16	fuji	Fuji Speedway	Oyama	Japan	35.3717	138.92700	583.0	http://en.wikipedia.org/wiki/Fuji_Speedway	1.000000
4	catalunya	Circuit de Barcelona-Catalunya	Montmeló	Spain	41.5700	2.26111	109.0	http://en.wikipedia.org/wiki/Circuit_de_Barcel...	2.800000
78	losail	Losail International Circuit	Al Daayen	Qatar	25.4900	51.45420	NaN	http://en.wikipedia.org/wiki/Losail_Internatio...	3.000000
14	monza	Autodromo Nazionale di Monza	Monza	Italy	45.6156	9.28111	162.0	http://en.wikipedia.org/wiki/Autodromo_Naziona...	3.222222
20	nurburgring	Nürburgring	Nürburg	Germany	50.3356	6.94750	578.0	http://en.wikipedia.org/wiki/N%C3%BCrburgring	3.250000

Opcional: Mostar todos los circuitos en un mapa con `folium`. En la libreta de ejercicios con *matplotlib* tenéis un ejemplo de cómo se hace.

In [58]:
```
!pip install folium
```

```
Requirement already satisfied: folium in c:\users\mbc98\anaconda3\lib\site-packages (0.13.0)
Requirement already satisfied: jinja2>=2.9 in c:\users\mbc98\anaconda3\lib\site-packages (from folium) (2.11.3)
Requirement already satisfied: requests in c:\users\mbc98\anaconda3\lib\site-packages (from folium) (2.27.1)
Requirement already satisfied: branca>=0.3.0 in c:\users\mbc98\anaconda3\lib\site-packages (from folium) (0.5.0)
Requirement already satisfied: numpy in c:\users\mbc98\anaconda3\lib\site-packages (from folium) (1.21.5)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\mbc98\anaconda3\lib\site-packages (from jinja2>=2.9->folium) (2.0.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\mbc98\anaconda3\lib\site-packages (from requests->folium) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\mbc98\anaconda3\lib\site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\mbc98\anaconda3\lib\site-packages (from requests->folium) (1.26.9)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\mbc98\anaconda3\lib\site-packages (from requests->folium) (2021.10.8)
```

In [59]:
```
import folium
```

In [60]:
```
##  Circuitos en los que Fernándo Alonso obtuvo, de media, la mejor posición.

tiles = 'OpenStreetMap'
world_figure = folium.Figure(width=600, height=400)
world_figure = folium.Map(location=[30, 0], tiles=tiles, zoom_start=2).add_to(world_figure)

# Fotos satélite
tiles = folium.TileLayer(
tiles = 'https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
attr = 'Esri',
name = 'Esri Satellite',
```

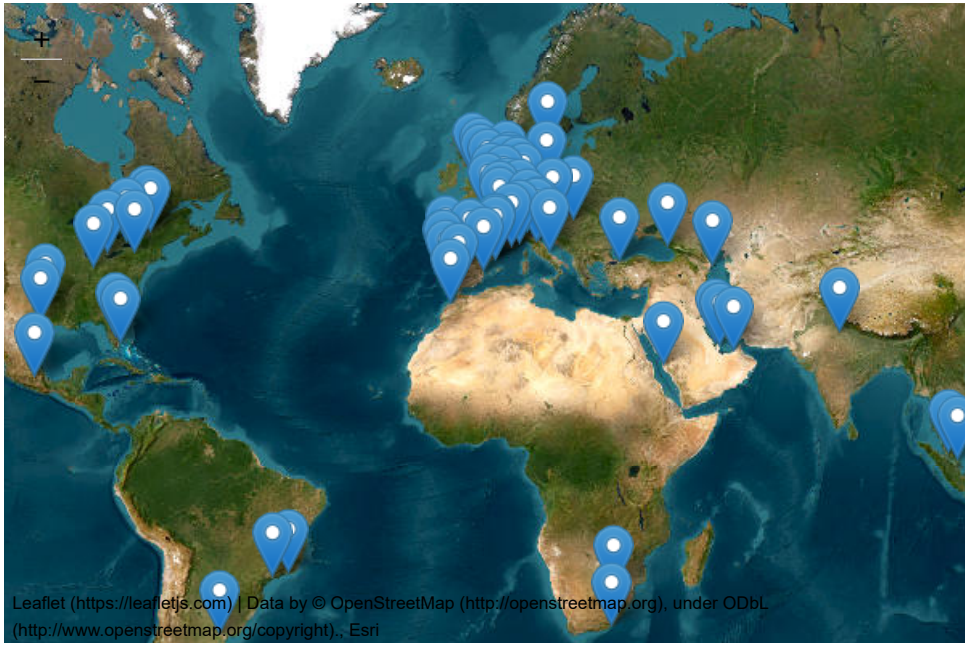
```
overlay = True,
control = True).add_to(world_figure)

for name,location in df_result_pos .iterrows():
    folium.Marker([location['lat'], location['lng']], popup=location['name']).add_to(world_figure)
world_figure
```

Out[60]:

In [61]:

```
## Todos los circuitos existentes en el mundo

tiles = 'OpenStreetMap'
world_figure = folium.Figure(width=600, height=400)
world_figure = folium.Map(location=[30, 0], tiles=tiles, zoom_start=2).add_to(world_figure)

# Fotos satélite
tiles = folium.TileLayer(
tiles = 'https://server.arcgisonline.com/ArcGIS/rest/services/World_Imagery/MapServer/tile/{z}/{y}/{x}',
attr = 'Esri',
name = 'Esri Satellite',
overlay = True,
control = True).add_to(world_figure)

for name,location in df_circuits.iterrows():
    folium.Marker([location['lat'], location['lng']], popup=location['name']).add_to(world_figure)
world_figure
```

Out[61]:



---

6. Alonso vs rivales

En esta parte del proyecto se valorarán los resultados de `Fernando Alonso` con respecto a sus rivales, es decir, con respecto a los pilotos que compitieron en las mismas carreras que él. Esto requiere manipular los datos, ya que la información no se puede obtener de forma directa.

Ejercicio 14

Mostrar los rivales (filas completas) contra los que ha competido Fernando Alonso.

▶

In [62]:
```
races_rivals = pd.DataFrame(set(df_results[ df_results['raceId']
                                        .isin(df_results_alo['raceId']]
                                              .values.tolist())]['driverId']), columns=['driverId'])
```

In [63]:
```
df_rivals_alonso = (pd.merge(races_rivals, df_drivers, left_on='driverId', right_on='driverId').rename_axis('id'))
df_rivals_alonso = df_rivals_alonso.loc[df_rivals_alonso['driverId']!= 4]
```

```
df_rivals_alonso.style.hide(axis='index')
```

Out[63]:

driverId	driverRef	number	code	forename	surname	dob	nationality	url
1	hamilton	44.000000	HAM	Lewis	Hamilton	1985-01-07 00:00:00	British	http://en.wikipedia.org/wiki/Lewis_Hamilton
2	heidfeld	nan	HEI	Nick	Heidfeld	1977-05-10 00:00:00	German	http://en.wikipedia.org/wiki/Nick_Heidfeld
3	rosberg	6.000000	ROS	Nico	Rosberg	1985-06-27 00:00:00	German	http://en.wikipedia.org/wiki/Nico_Rosberg
5	kovalainen	nan	KOV	Heikki	Kovalainen	1981-10-19 00:00:00	Finnish	http://en.wikipedia.org/wiki/Heikki_Kovalainen
6	nakajima	nan	NAK	Kazuki	Nakajima	1985-01-11 00:00:00	Japanese	http://en.wikipedia.org/wiki/Kazuki_Nakajima
7	bourdais	nan	BOU	Sébastien	Bourdais	1979-02-28 00:00:00	French	http://en.wikipedia.org/wiki/S%C3%A9bastien_Bourdais
8	raikkonen	7.000000	RAI	Kimi	Räikkönen	1979-10-17 00:00:00	Finnish	http://en.wikipedia.org/wiki/Kimi_R%C3%A4ikk%C3%B6nen
9	kubica	88.000000	KUB	Robert	Kubica	1984-12-07 00:00:00	Polish	http://en.wikipedia.org/wiki/Robert_Kubica
10	glock	nan	GLO	Timo	Glock	1982-03-18 00:00:00	German	http://en.wikipedia.org/wiki/Timo_Glock
11	sato	nan	SAT	Takuma	Sato	1977-01-28 00:00:00	Japanese	http://en.wikipedia.org/wiki/Takuma_Sato
12	piquet_jr	nan	PIQ	Nelson	Piquet Jr.	1985-07-25 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Nelson_Piquet,_Jr.
13	massa	19.000000	MAS	Felipe	Massa	1981-04-25 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Felipe_Massa
14	coulthard	nan	COU	David	Coulthard	1971-03-27 00:00:00	British	http://en.wikipedia.org/wiki/David_Coulthard
15	trulli	nan	TRU	Jarno	Trulli	1974-07-13 00:00:00	Italian	http://en.wikipedia.org/wiki/Jarno_Trulli
16	sutil	99.000000	SUT	Adrian	Sutil	1983-01-11 00:00:00	German	http://en.wikipedia.org/wiki/Adrian_Sutil
17	webber	nan	WEB	Mark	Webber	1976-08-27 00:00:00	Australian	http://en.wikipedia.org/wiki/Mark_Webber_(racing_driver)
18	button	22.000000	BUT	Jenson	Button	1980-01-19 00:00:00	British	http://en.wikipedia.org/wiki/Jenson_Button
19	davidson	nan	DAV	Anthony	Davidson	1979-04-18 00:00:00	British	http://en.wikipedia.org/wiki/Anthony_Davidson
20	vettel	5.000000	VET	Sebastian	Vettel	1987-07-03 00:00:00	German	http://en.wikipedia.org/wiki/Sebastian_Vettel
21	fisichella	nan	FIS	Giancarlo	Fisichella	1973-01-14 00:00:00	Italian	http://en.wikipedia.org/wiki/Giancarlo_Fisichella
22	barrichello	nan	BAR	Rubens	Barrichello	1972-05-23 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Rubens_Barrichello
23	ralf_schumacher	nan	SCH	Ralf	Schumacher	1975-06-30 00:00:00	German	http://en.wikipedia.org/wiki/Ralf_Schumacher
24	liuzzi	nan	LIU	Vitantonio	Liuzzi	1980-08-06 00:00:00	Italian	http://en.wikipedia.org/wiki/Vitantonio_Liuzzi
25	wurz	nan	WUR	Alexander	Wurz	1974-02-15 00:00:00	Austrian	http://en.wikipedia.org/wiki/Alexander_Wurz
26	speed	nan	SPE	Scott	Speed	1983-01-24 00:00:00	American	http://en.wikipedia.org/wiki/Scott_Speed
27	albers	nan	ALB	Christijan	Albers	1979-04-16 00:00:00	Dutch	http://en.wikipedia.org/wiki/Christijan_Albers
28	markus_winkelhock	nan	WIN	Markus	Winkelhock	1980-06-13 00:00:00	German	http://en.wikipedia.org/wiki/Markus_Winkelhock
29	yamamoto	nan	YAM	Sakon	Yamamoto	1982-07-09 00:00:00	Japanese	http://en.wikipedia.org/wiki/Sakon_Yamamoto
30	michael_schumacher	nan	MSC	Michael	Schumacher	1969-01-03 00:00:00	German	http://en.wikipedia.org/wiki/Michael_Schumacher
31	montoya	nan	MON	Juan	Pablo Montoya	1975-09-20 00:00:00	Colombian	http://en.wikipedia.org/wiki/Juan_Pablo_Montoya
32	klien	nan	KLI	Christian	Klien	1983-02-07 00:00:00	Austrian	http://en.wikipedia.org/wiki/Christian_Klien

driverId	driverRef	number	code	forename	surname	dob	nationality	url
33	monteiro	nan	TMO	Tiago	Monteiro	1976-07-24 00:00:00	Portuguese	http://en.wikipedia.org/wiki/Tiago_Monteiro
34	ide	nan	IDE	Yuji	Ide	1975-01-21 00:00:00	Japanese	http://en.wikipedia.org/wiki/Yuji_Ide
35	villeneuve	nan	VIL	Jacques	Villeneuve	1971-04-09 00:00:00	Canadian	http://en.wikipedia.org/wiki/Jacques_Villeneuve
36	montagny	nan	FMO	Franck	Montagny	1978-01-05 00:00:00	French	http://en.wikipedia.org/wiki/Franck_Montagny
37	rosa	nan	DLR	Pedro	de la Rosa	1971-02-24 00:00:00	Spanish	http://en.wikipedia.org/wiki/Pedro_de_la_Rosa
38	doornbos	nan	DOO	Robert	Doornbos	1981-09-23 00:00:00	Dutch	http://en.wikipedia.org/wiki/Robert_Doornbos
39	karthikeyan	nan	KAR	Narain	Karthikeyan	1977-01-14 00:00:00	Indian	http://en.wikipedia.org/wiki/Narain_Karthikeyan
40	friesacher	nan	FRI	Patrick	Friesacher	1980-09-26 00:00:00	Austrian	http://en.wikipedia.org/wiki/Patrick_Friesacher
41	zonta	nan	ZON	Ricardo	Zonta	1976-03-23 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Ricardo_Zonta
42	pizzonia	nan	PIZ	Antônio	Pizzonia	1980-09-11 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Ant%C3%B4nio_Pizzonia
43	matta	nan		Cristiano	da Matta	1973-09-19 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Cristiano_da_Matta
44	panis	nan		Olivier	Panis	1966-09-02 00:00:00	French	http://en.wikipedia.org/wiki/Olivier_Panis
45	pantano	nan		Giorgio	Pantano	1979-02-04 00:00:00	Italian	http://en.wikipedia.org/wiki/Giorgio_Pantano
46	bruni	nan		Gianmaria	Bruni	1981-05-30 00:00:00	Italian	http://en.wikipedia.org/wiki/Gianmaria_Bruni
47	baumgartner	nan		Zsolt	Baumgartner	1981-01-01 00:00:00	Hungarian	http://en.wikipedia.org/wiki/Zsolt_Baumgartner
48	gene	nan		Marc	Gené	1974-03-29 00:00:00	Spanish	http://en.wikipedia.org/wiki/Marc_Gen%C3%A9
49	frentzen	nan		Heinz-Harald	Frentzen	1967-05-18 00:00:00	German	http://en.wikipedia.org/wiki/Heinz-Harald_Frentzen
50	verstappen	nan		Jos	Verstappen	1972-03-04 00:00:00	Dutch	http://en.wikipedia.org/wiki/Jos_Verstappen
51	wilson	nan		Justin	Wilson	1978-07-31 00:00:00	British	http://en.wikipedia.org/wiki/Justin_Wilson_(racing_driver)
52	firman	nan		Ralph	Firman	1975-05-20 00:00:00	Irish	http://en.wikipedia.org/wiki/Ralph_Firman
53	kiesa	nan		Nicolas	Kiesa	1978-03-03 00:00:00	Danish	http://en.wikipedia.org/wiki/Nicolas_Kiesa
67	buemi	nan	BUE	Sébastien	Buemi	1988-10-31 00:00:00	Swiss	http://en.wikipedia.org/wiki/S%C3%A9bastien_Buemi
69	badoer	nan	BAD	Luca	Badoer	1971-01-25 00:00:00	Italian	http://en.wikipedia.org/wiki/Luca_Badoer
153	alguersuari	nan	ALG	Jaime	Alguersuari	1990-03-23 00:00:00	Spanish	http://en.wikipedia.org/wiki/Jaime_Alguersuari
154	grosjean	8.000000	GRO	Romain	Grosjean	1986-04-17 00:00:00	French	http://en.wikipedia.org/wiki/Romain_Grosjean
155	kobayashi	10.000000	KOB	Kamui	Kobayashi	1986-09-13 00:00:00	Japanese	http://en.wikipedia.org/wiki/Kamui_Kobayashi
807	hulkenberg	27.000000	HUL	Nico	Hülkenberg	1987-08-19 00:00:00	German	http://en.wikipedia.org/wiki/Nico_H%C3%BClkenberg
808	petrov	nan	PET	Vitaly	Petrov	1984-09-08 00:00:00	Russian	http://en.wikipedia.org/wiki/Vitaly_Petrov
810	grassi	nan	DIG	Lucas	di Grassi	1984-08-11 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Lucas_di_Grassi
811	bruno_senna	nan	SEN	Bruno	Senna	1983-10-15 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Bruno_Senna
812	chandhok	nan	CHA	Karun	Chandhok	1984-01-19 00:00:00	Indian	http://en.wikipedia.org/wiki/Karun_Chandhok

driverId	driverRef	number	code	forename	surname	dob	nationality	url
813	maldonado	13.000000	MAL	Pastor	Maldonado	1985-03-09 00:00:00	Venezuelan	http://en.wikipedia.org/wiki/Pastor_Maldonado
814	resta	nan	DIR	Paul	di Resta	1986-04-16 00:00:00	British	http://en.wikipedia.org/wiki/Paul_di_Resta
815	perez	11.000000	PER	Sergio	Pérez	1990-01-26 00:00:00	Mexican	http://en.wikipedia.org/wiki/Sergio_P%C3%A9rez
816	ambrosio	nan	DAM	Jérôme	d'Ambrosio	1985-12-27 00:00:00	Belgian	http://en.wikipedia.org/wiki/J%C3%A9r%C3%B4me_d%27Ambrosio
817	ricciardo	3.000000	RIC	Daniel	Ricciardo	1989-07-01 00:00:00	Australian	http://en.wikipedia.org/wiki/Daniel_Ricciardo
818	vergne	25.000000	VER	Jean-Éric	Vergne	1990-04-25 00:00:00	French	http://en.wikipedia.org/wiki/Jean-%C3%89ric_Vergne
819	pic	nan	PIC	Charles	Pic	1990-02-15 00:00:00	French	http://en.wikipedia.org/wiki/Charles_Pic
820	chilton	4.000000	CHI	Max	Chilton	1991-04-21 00:00:00	British	http://en.wikipedia.org/wiki/Max_Chilton
821	gutierrez	21.000000	GUT	Esteban	Gutiérrez	1991-08-05 00:00:00	Mexican	http://en.wikipedia.org/wiki/Esteban_Guti%C3%A9rrez
822	bottas	77.000000	BOT	Valtteri	Bottas	1989-08-28 00:00:00	Finnish	http://en.wikipedia.org/wiki/Valtteri_Bottas
823	garde	nan	VDG	Giedo	van der Garde	1985-04-25 00:00:00	Dutch	http://en.wikipedia.org/wiki/Giedo_van_der_Garde
824	jules_bianchi	17.000000	BIA	Jules	Bianchi	1989-08-03 00:00:00	French	http://en.wikipedia.org/wiki/Jules_Bianchi
825	kevin_magnussen	20.000000	MAG	Kevin	Magnussen	1992-10-05 00:00:00	Danish	http://en.wikipedia.org/wiki/Kevin_Magnussen
826	kvyat	26.000000	KVY	Daniil	Kvyat	1994-04-26 00:00:00	Russian	http://en.wikipedia.org/wiki/Daniil_Kvyat
827	lotterer	45.000000	LOT	André	Lotterer	1981-11-19 00:00:00	German	http://en.wikipedia.org/wiki/Andr%C3%A9_Lotterer
828	ericsson	9.000000	ERI	Marcus	Ericsson	1990-09-02 00:00:00	Swedish	http://en.wikipedia.org/wiki/Marcus_Ericsson
829	stevens	28.000000	STE	Will	Stevens	1991-06-28 00:00:00	British	http://en.wikipedia.org/wiki/Will_Stevens
830	max_verstappen	33.000000	VER	Max	Verstappen	1997-09-30 00:00:00	Dutch	http://en.wikipedia.org/wiki/Max_Verstappen
831	nasr	12.000000	NAS	Felipe	Nasr	1992-08-21 00:00:00	Brazilian	http://en.wikipedia.org/wiki/Felipe_Nasr
832	sainz	55.000000	SAI	Carlos	Sainz	1994-09-01 00:00:00	Spanish	http://en.wikipedia.org/wiki/Carlos_Sainz_Jr.
833	merhi	98.000000	MER	Roberto	Merhi	1991-03-22 00:00:00	Spanish	http://en.wikipedia.org/wiki/Roberto_Merhi
834	rossi	53.000000	RSS	Alexander	Rossi	1991-09-25 00:00:00	American	http://en.wikipedia.org/wiki/Alexander_Rossi_%28racing_driver%29
835	jolyon_palmer	30.000000	PAL	Jolyon	Palmer	1991-01-20 00:00:00	British	http://en.wikipedia.org/wiki/Jolyon_Palmer
836	wehrlein	94.000000	WEH	Pascal	Wehrlein	1994-10-18 00:00:00	German	http://en.wikipedia.org/wiki/Pascal_Wehrlein
837	haryanto	88.000000	HAR	Rio	Haryanto	1993-01-22 00:00:00	Indonesian	http://en.wikipedia.org/wiki/Rio_Haryanto
838	vandoorne	2.000000	VAN	Stoffel	Vandoorne	1992-03-26 00:00:00	Belgian	http://en.wikipedia.org/wiki/Stoffel_Vandoorne
839	ocon	31.000000	OCO	Esteban	Ocon	1996-09-17 00:00:00	French	http://en.wikipedia.org/wiki/Esteban_Ocon
840	stroll	18.000000	STR	Lance	Stroll	1998-10-29 00:00:00	Canadian	http://en.wikipedia.org/wiki/Lance_Stroll
841	giovinazzi	99.000000	GIO	Antonio	Giovinazzi	1993-12-14 00:00:00	Italian	http://en.wikipedia.org/wiki/Antonio_Giovinazzi
842	gasly	10.000000	GAS	Pierre	Gasly	1996-02-07 00:00:00	French	http://en.wikipedia.org/wiki/Pierre_Gasly
843	brendon_hartley	28.000000	HAR	Brendon	Hartley	1989-11-10 00:00:00	New Zealander	http://en.wikipedia.org/wiki/Brendon_Hartley

driverId	driverRef	number	code	forename	surname	dob	nationality	url
844	leclerc	16.000000	LEC	Charles	Leclerc	1997-10-16 00:00:00	Monegasque	http://en.wikipedia.org/wiki/Charles_Leclerc
845	sirotkin	35.000000	SIR	Sergey	Sirotkin	1995-08-27 00:00:00	Russian	http://en.wikipedia.org/wiki/Sergey_Sirotkin_(racing_driver)
846	norris	4.000000	NOR	Lando	Norris	1999-11-13 00:00:00	British	http://en.wikipedia.org/wiki/Lando_Norris
847	russell	63.000000	RUS	George	Russell	1998-02-15 00:00:00	British	http://en.wikipedia.org/wiki/George_Russell_%28racing_driver%29
848	albon	23.000000	ALB	Alexander	Albon	1996-03-23 00:00:00	Thai	http://en.wikipedia.org/wiki/Alexander_Albon
849	latifi	6.000000	LAT	Nicholas	Latifi	1995-06-29 00:00:00	Canadian	http://en.wikipedia.org/wiki/Nicholas_Latifi
852	tsunoda	22.000000	TSU	Yuki	Tsunoda	2000-05-11 00:00:00	Japanese	http://en.wikipedia.org/wiki/Yuki_Tsunoda
853	mazepin	9.000000	MAZ	Nikita	Mazepin	1999-03-02 00:00:00	Russian	http://en.wikipedia.org/wiki/Nikita_Mazepin
854	mick_schumacher	47.000000	MSC	Mick	Schumacher	1999-03-22 00:00:00	German	http://en.wikipedia.org/wiki/Mick_Schumacher
855	zhou	24.000000	ZHO	Guanyu	Zhou	1999-05-30 00:00:00	Chinese	http://en.wikipedia.org/wiki/Guanyu_Zhou

Obtener los datos de los 10 rivales a los que más veces se ha enfrentado `Fernando Alonso`, y almacenarlos en el *DataFrame* `df_fa_vs`.

▶

In [64]:
```
list_races_alonso = set(df_results.loc[(df_results['driverId']==4)]['raceId'])
#list_races_alonso
```

In [65]:
```
id_rivals_alonso = df_results[ df_results['raceId'].isin(list_races_alonso) ]
id_rivals_alonso = id_rivals_alonso.loc[id_rivals_alonso['driverId']!= 4].groupby('driverId')['driverId'].count()
id_rivals_alonso = id_rivals_alonso.nlargest(10).to_frame(name= 'amount_races')
#id_rivals_alonso
```

In [66]:
```
df_fa_vs = pd.merge(id_rivals_alonso, df_drivers, left_on='driverId', right_on='driverId')
df_fa_vs
```

Out[66]:

driverId	amount_races	driverRef	number	code	forename	surname	dob	nationality	url
8	294	raikkonen	7.0	RAI	Kimi	Räikkönen	1979-10-17	Finnish	http://en.wikipedia.org/wiki/Kimi_R%C3%A4ikk%C...
18	272	button	22.0	BUT	Jenson	Button	1980-01-19	British	http://en.wikipedia.org/wiki/Jenson_Button
1	261	hamilton	44.0	HAM	Lewis	Hamilton	1985-01-07	British	http://en.wikipedia.org/wiki/Lewis_Hamilton
13	252	massa	19.0	MAS	Felipe	Massa	1981-04-25	Brazilian	http://en.wikipedia.org/wiki/Felipe_Massa
20	250	vettel	5.0	VET	Sebastian	Vettel	1987-07-03	German	http://en.wikipedia.org/wiki/Sebastian_Vettel
3	204	rosberg	6.0	ROS	Nico	Rosberg	1985-06-27	German	http://en.wikipedia.org/wiki/Nico_Rosberg
17	200	webber	NaN	WEB	Mark	Webber	1976-08-27	Australian	http://en.wikipedia.org/wiki/Mark_Webber_(raci...
815	189	perez	11.0	PER	Sergio	Pérez	1990-01-26	Mexican	http://en.wikipedia.org/wiki/Sergio_P%C3%A9rez
817	182	ricciardo	3.0	RIC	Daniel	Ricciardo	1989-07-01	Australian	http://en.wikipedia.org/wiki/Daniel_Ricciardo
22	178	barrichello	NaN	BAR	Rubens	Barrichello	1972-05-23	Brazilian	http://en.wikipedia.org/wiki/Rubens_Barrichello

A continuación se hará un balance de los resultados de `Fernando Alonso` con respecto a sus rivales. Para ello, es necesario pasar por una serie de pasos intermedios.

Ejercicio 15

Obtener una tabla en la que cada fila represente cada una de las carreras (`raceId`) concluidas por `Fernando Alonso` , cada columna un piloto (`driverId`) que haya participado (y concluido) en alguna de esas carreras, y el resultado la posición (`position`) del piloto. Almacenar el resultado en un *DataFrame* denominado `df_fa_vs_other` .

▶

In [67]:
```
# Obtengo la lista de las carreras terminadas de Fernándo Alonso
races_alonso = df_results_alo['raceId'].values.tolist()
#races_alonso
```

In [68]:
```
# Obtengo la lista de las carreras terminadas de los rivales de Fernándo Alonso
df_races_rivals = df_results.loc[(df_results['driverId']!= 4) & (df_results['statusId']== 1)]
#df_races_rivals
```

In [69]:
```
# Escojo las carreras en las que corrió Alonso
df_races_rivals  = df_races_rivals [df_races_rivals ['raceId'].isin(races_alonso)][['raceId','driverId','position']].set_index(['raceId','driverId'])

# Creo la tabla
df_fa_vs_other = df_races_rivals.unstack(level=1)
#df_fa_vs_other
```

In [70]:
```
# Para ver mejor la tabla, sustituyo NaN por -
df_fa_vs_other_pretty = df_races_rivals.unstack(level=1, fill_value='-')
df_fa_vs_other_pretty
```

Out[70]:

	position																				
driverId	**1**	**2**	**3**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**...**	**844**	**845**	**846**	**847**	**848**	**849**	**852**	**853**	**854**	**855**
raceId																					
1	-	10.0	6.0	-	-	8.0	-	-	4.0	-	...	-	-	-	-	-	-	-	-	-	-
3	6.0	12.0	-	5.0	-	11.0	10.0	13.0	7.0	-	...	-	-	-	-	-	-	-	-	-	-
4	4.0	-	9.0	12.0	-	13.0	6.0	-	7.0	-	...	-	-	-	-	-	-	-	-	-	-
5	-	7.0	8.0	-	-	-	-	-	-	-	...	-	-	-	-	-	-	-	-	-	-
6	-	-	6.0	-	-	8.0	3.0	-	-	-	...	-	-	-	-	-	-	-	-	-	-
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1080	8.0	-	-	-	-	-	-	-	-	-	...	4.0	-	6.0	5.0	-	-	-	-	-	-
1081	4.0	-	-	-	-	-	-	-	-	-	...	-	-	9.0	3.0	-	-	-	-	-	-
1082	3.0	-	-	-	-	-	-	-	-	-	...	5.0	-	15.0	4.0	13.0	16.0	-	-	-	8.0
1083	3.0	-	-	-	-	-	-	-	-	-	...	4.0	-	6.0	-	-	12.0	14.0	-	8.0	-
1085	2.0	-	-	-	-	-	-	-	-	-	...	-	-	7.0	3.0	13.0	-	-	-	15.0	-

220 rows × 78 columns

Ejercicio 16

A partir del *DataFrame* `df_fa_vs_other` mostrar los 10 rivales ha batido más veces `Fernando Alonso`.

▶

In [71]:
```
df_alonso_victories = (df_fa_vs_other.apply(lambda df_row: df_row > df_row[4], axis=1)
                       .fillna(0)
                       .sum(axis = 0)
                       .nlargest(10)
                       .to_frame()
                       .reset_index()
                       .rename(columns={0:'times_alonso_won'})
                       .merge(df_drivers, on="driverId", how='inner'))

df_alonso_victories
```

Out[71]:

	level_0	driverId	times_alonso_won	driverRef	number	code	forename	surname	dob	nationality	url
0	position	15	3	trulli	NaN	TRU	Jarno	Trulli	1974-07-13	Italian	http://en.wikipedia.org/wiki/Jarno_Trulli
1	position	21	3	fisichella	NaN	FIS	Giancarlo	Fisichella	1973-01-14	Italian	http://en.wikipedia.org/wiki/Giancarlo_Fisichella
2	position	3	2	rosberg	6.0	ROS	Nico	Rosberg	1985-06-27	German	http://en.wikipedia.org/wiki/Nico_Rosberg
3	position	7	2	bourdais	NaN	BOU	Sébastien	Bourdais	1979-02-28	French	http://en.wikipedia.org/wiki/S%C3%A9bastien_Bo...
4	position	9	2	kubica	88.0	KUB	Robert	Kubica	1984-12-07	Polish	http://en.wikipedia.org/wiki/Robert_Kubica
5	position	16	2	sutil	99.0	SUT	Adrian	Sutil	1983-01-11	German	http://en.wikipedia.org/wiki/Adrian_Sutil
6	position	18	2	button	22.0	BUT	Jenson	Button	1980-01-19	British	http://en.wikipedia.org/wiki/Jenson_Button
7	position	24	2	liuzzi	NaN	LIU	Vitantonio	Liuzzi	1980-08-06	Italian	http://en.wikipedia.org/wiki/Vitantonio_Liuzzi
8	position	1	1	hamilton	44.0	HAM	Lewis	Hamilton	1985-01-07	British	http://en.wikipedia.org/wiki/Lewis_Hamilton
9	position	5	1	kovalainen	NaN	KOV	Heikki	Kovalainen	1981-10-19	Finnish	http://en.wikipedia.org/wiki/Heikki_Kovalainen

Mostrar ahora los 10 rivales han batido más veces a `Fernando Alonso`.

▶

In [72]:
```
df_alonso_defeats = (df_fa_vs_other.apply(lambda df_row: df_row < df_row[4], axis=1)
                     .fillna(0)
                     .sum(axis = 0)
                     .nlargest(10)
                     .to_frame()
                     .reset_index()
                     .rename(columns={0:'times_alonso_lost'})
                     .merge(df_drivers, on="driverId", how='inner'))

df_alonso_defeats
```

Out[72]:

	level_0	driverId	times_alonso_lost	driverRef	number	code	forename	surname	dob	nationality	url
0	position	20	10	vettel	5.0	VET	Sebastian	Vettel	1987-07-03	German	http://en.wikipedia.org/wiki/Sebastian_Vettel
1	position	2	9	heidfeld	NaN	HEI	Nick	Heidfeld	1977-05-10	German	http://en.wikipedia.org/wiki/Nick_Heidfeld
2	position	5	7	kovalainen	NaN	KOV	Heikki	Kovalainen	1981-10-19	Finnish	http://en.wikipedia.org/wiki/Heikki_Kovalainen
3	position	8	7	raikkonen	7.0	RAI	Kimi	Räikkönen	1979-10-17	Finnish	http://en.wikipedia.org/wiki/Kimi_R%C3%A4ikk%C...
4	position	9	7	kubica	88.0	KUB	Robert	Kubica	1984-12-07	Polish	http://en.wikipedia.org/wiki/Robert_Kubica
5	position	1	6	hamilton	44.0	HAM	Lewis	Hamilton	1985-01-07	British	http://en.wikipedia.org/wiki/Lewis_Hamilton
6	position	3	6	rosberg	6.0	ROS	Nico	Rosberg	1985-06-27	German	http://en.wikipedia.org/wiki/Nico_Rosberg
7	position	10	6	glock	NaN	GLO	Timo	Glock	1982-03-18	German	http://en.wikipedia.org/wiki/Timo_Glock
8	position	18	6	button	22.0	BUT	Jenson	Button	1980-01-19	British	http://en.wikipedia.org/wiki/Jenson_Button
9	position	22	6	barrichello	NaN	BAR	Rubens	Barrichello	1972-05-23	Brazilian	http://en.wikipedia.org/wiki/Rubens_Barrichello

---

7. Conclusiones y líneas de trabajo para TFM

Este trabajo ha servido para familiarizarnos con la herramienta *pandas* y hacer algunas operaciones algo más complejas orientadas al análisis. Aunque se trata de un primer proyecto de carácter *instrumental*, desde el punto de vista de los datos nos ha permitido además obtener información en principio desconocida, o al menos en parte.

Como puede apreciarse, el conjunto de datos es más amplio, y contiene información sobre tiempos por vuelta, temporadas, etc. Si os interesa o apetece, es posible extender el análisis que se ha hecho para que contemple esas y otras cuestiones.

Líneas de trabajo para TFM

Con respecto al TFM, existe la posibilidad de hacer un portfolio con extensiones o trabajo adicional relacionado con los *Capstones* del máster, incluyendo varios de ellos elegidos librermente. El trabajo adicional que pudiéseis hacer aquí podría estar enfocado hacia ese objetivo, y formar parte del portfolio.

En ese sentido apuntar que, por lo general, el trabajo se acumula conforme transcurre el año, que el último proyecto se os da en mayo, y que quizá *algunos* prefiráis ir avanzando en un portfolio antes que comenzar un TFM al final.
