```
import pandas as pd
```

Importar Datos

```
station_col_names = ['id', 'station', 'municipality', 'lat', 'lng']

station_df = pd.read_csv("/content/drive/MyDrive/data_stations.txt", names=station_col_names, header=None, delimiter=',', quotecha
station_df
```

	id	station	municipality	lat	lng
0	3	Colleges of the Fenway	Boston	42.340021	-71.100812
1	4	Tremont St. at Berkeley St.	Boston	42.345392	-71.069616
2	5	Northeastern U / North Parking Lot	Boston	42.341814	-71.090179
3	6	Cambridge St. at Joy St.	Boston	42.361285	-71.065140
4	7	Fan Pier	Boston	42.353412	-71.044624
...	...	...	...	...	...
132	141	Powder House Circle	Somerville	42.400877	-71.116772
133	142	Packard Ave / Powderhouse Blvd	Somerville	42.404490	-71.123413
134	143	Somerville Hospital at Highland Ave / Crocker St	Somerville	42.390820	-71.109420
135	144	Teele Square at 239 Holland St	Somerville	42.402763	-71.126908
136	145	Summer St at Cutter St	Somerville	42.394002	-71.120406

137 rows × 5 columns

Pasos siguientes: Generar código con `station_df` Ver gráficos recomendados

```
trips_col_names = ['id', 'duration', 'start_date', 'start_station', 'end_date', 'end_station', 'bike_number', 'sub_type', 'zip_coc

trips_df = pd.read_csv('/content/drive/MyDrive/data_trips.txt', names=trips_col_names, header=None, delimiter=',', quotechar="'",
trips_df
```

```
<ipython-input-67-f2c3ba403947>:3: DtypeWarning: Columns (8) have mixed types. Specify dtype option on import or set lo
  trips_df = pd.read_csv('/content/drive/MyDrive/data_trips.txt', names=trips_col_names, header=None, delimiter=',', qu
```

	id	duration	start_date	start_station	end_date	end_station	bike_number	sub_type	zip_code	birth_da
0	1	9	2011-07-28 10:12:00	99.0	2011-07-28 10:12:00	99.0	B00468	Registered	97217.0	197
1	2	220	2011-07-28 10:21:00	99.0	2011-07-28 10:25:00	99.0	B00554	Registered	2215.0	196
2	3	100	2011-07-28 10:33:00	99.0	2011-07-28 10:34:00	99.0	B00456	Registered	2108.0	194
3	4	64	2011-07-28 10:35:00	99.0	2011-07-28 10:36:00	99.0	B00554	Registered	2116.0	198
4	5	12	2011-07-28 10:37:00	99.0	2011-07-28 10:37:00	99.0	B00554	Registered	97214.0	198
...	...	...	...	...	...	...	...	...	...	
1566952	1579021	720	2013-11-30 23:30:00	130.0	2013-11-30 23:42:00	90.0	T01341	Registered	02141	
1566953	1579022	480	2013-11-30 23:32:00	67.0	2013-11-30 23:40:00	88.0	T01328	Registered	02143	
1566954	1579023	540	2013-11-30 23:32:00	137.0	2013-11-30 23:41:00	133.0	T01310	Casual	NaN	N

Limpiar Datos

```
# Comprobar que los valores de 'sub_type' solo son 'Registered' y 'Casual'
trips_df['sub_type'].unique()
```

```
array(['Registered', 'Casual'], dtype=object)
```

```
# Rellenar los valores Nan con 'Not Known'
trips_df['gender'] = trips_df['gender'].fillna('Not Known')
```

```
# Sustituir las entradas con valor 'Male ' por 'Male'
trips_df['gender'] = trips_df['gender'].replace('Male ', 'Male')
```

```
# Comprobar que los valores de 'gender' solo son 'Male' y 'Female'
trips_df['gender'].unique()
```

```
array(['Male', 'Female', 'Not Known'], dtype=object)
```

```
# Convertir fechas
trips_df['start_date'] = pd.to_datetime(trips_df['start_date'])
trips_df['end_date'] = pd.to_datetime(trips_df['end_date'])
```

```
# Eliminar bicicletas sin codigo
trips_df = trips_df[trips_df['bike_number'] != ' ']
```

Normalizar Datos

```
bikes_df = trips_df[['bike_number']].drop_duplicates().reset_index(drop=True)
bikes_df
```

	bike_number
0	B00468
1	B00554
2	B00456
3	B00550
4	B00580
...	...
1159	B00642
1160	B00652
1161	B00630
1162	B00624
1163	B01491

1164 rows × 1 columns

Pasos siguientes: Generar código con `bikes_df` Ver gráficos recomendados

```
people_df = trips_df[['zip_code', 'birth_date', 'gender']].drop_duplicates().reset_index(drop=True)
people_df
```

	zip_code	birth_date	gender
0	97217.0	1976.0	Male
1	2215.0	1966.0	Male
2	2108.0	1943.0	Male
3	2116.0	1981.0	Female
4	97214.0	1983.0	Female
...	...	...	...
4454	02121	0.0	Female
4455	01741	0.0	Male
4456	03823	0.0	Female
4457	02493	0.0	Female
4458	y02143	0.0	Female

4459 rows × 3 columns

Pasos siguientes: Generar código con `people_df` Ver gráficos recomendados

```
trips_df = trips_df.merge(people_df, on=['zip_code', 'birth_date', 'gender'], how='left', suffixes=('', '_people'))
trips_df
```

	id	duration	start_date	start_station	end_date	end_station	bike_number	sub_type	zip_code	birth_da
0	1	9	2011-07-28 10:12:00	99.0	2011-07-28 10:12:00	99.0	B00468	Registered	97217.0	197
1	2	220	2011-07-28 10:21:00	99.0	2011-07-28 10:25:00	99.0	B00554	Registered	2215.0	196
2	3	100	2011-07-28 10:33:00	99.0	2011-07-28 10:34:00	99.0	B00456	Registered	2108.0	194
3	4	64	2011-07-28 10:35:00	99.0	2011-07-28 10:36:00	99.0	B00554	Registered	2116.0	198
4	5	12	2011-07-28 10:37:00	99.0	2011-07-28 10:37:00	99.0	B00554	Registered	97214.0	198
...	...	...	...	...	...	...	...	...	...	
1566951	1579021	720	2013-11-30 23:30:00	130.0	2013-11-30 23:42:00	90.0	T01341	Registered	02141	
1566952	1579022	480	2013-11-30 23:32:00	67.0	2013-11-30 23:40:00	88.0	T01328	Registered	02143	
1566953	1579023	540	2013-11-30	137.0	2013-11-30	133.0	T01310	Casual	NaN	N

```
trips_df = trips_df.merge(bikes_df, on='bike_number', how='left', suffixes=('', '_bike'))
trips_df
```

	id	duration	start_date	start_station	end_date	end_station	bike_number	sub_type	zip_code	birth_da
0	1	9	2011-07-28 10:12:00	99.0	2011-07-28 10:12:00	99.0	B00468	Registered	97217.0	197
1	2	220	2011-07-28 10:21:00	99.0	2011-07-28 10:25:00	99.0	B00554	Registered	2215.0	196
2	3	100	2011-07-28 10:33:00	99.0	2011-07-28 10:34:00	99.0	B00456	Registered	2108.0	194
3	4	64	2011-07-28 10:35:00	99.0	2011-07-28 10:36:00	99.0	B00554	Registered	2116.0	198
4	5	12	2011-07-28 10:37:00	99.0	2011-07-28 10:37:00	99.0	B00554	Registered	97214.0	198
...	...	...	...	...	...	...	...	...	...	
1566951	1579021	720	2013-11-30 23:30:00	130.0	2013-11-30 23:42:00	90.0	T01341	Registered	02141	
1566952	1579022	480	2013-11-30 23:32:00	67.0	2013-11-30 23:40:00	88.0	T01328	Registered	02143	
1566953	1579023	540	2013-11-30	137.0	2013-11-30	133.0	T01310	Casual	NaN	N

Métricas

1. ¿Cual es la media de la duración de los viajes? ¿ Numero total de trayectos ?
2. Minutos de bicileta segun edad del cliente
3. Y si tienes en cuenta sólo los viajes reales (supongamos que son los que duran más de 1 minuto) ¿Cuántas bicicletas hay registradas?
4. Distribucion del grado de obsolescencia del parque de bicicletas , considerando que la vida util es de 1.800 viajes
5. ¿Puedes mostrarme una tabla con las bicicletas y el número de viajes que han realizado?
6. ¿Puedes mostrarme una tabla con las 10 bicicletas con más viajes nulos realizados (los de menos de 60 segundos?

7. ¿Cual es la bicicleta que más se ha usado segun las edades de los conductores?
8. ¿Qué bicicletas han sido usadas en más de 2.000 viajes de al menos 3 minutos?
9. Análisis temporal de los datos

```
# 1
mean_trip = trips_df['duration'].mean()
print("Duración media (minutos):", mean_trip)

total_trips = trips_df['id'].count()
print("Número total de trayectos:", total_trips)
```

```
Duración media (minutos): 911.5683497175415
Número total de trayectos: 1566956
```

```
# 2
from datetime import datetime

current_year = datetime.now().year

ages_df = trips_df[['birth_date', 'duration']]

ages_df = ages_df.dropna()
ages_df = ages_df[ages_df['birth_date'] > 0]
ages_df['age'] = current_year - ages_df['birth_date']
ages_df = ages_df.drop(columns=['birth_date'])

minutes_by_age = ages_df.groupby('age')['duration'].sum().reset_index()
minutes_by_age
```

28	57.0	3952288
29	58.0	4031539
30	59.0	3520872
31	60.0	2750997
32	61.0	3811476
33	62.0	3990603
34	63.0	3008824
35	64.0	2987984
36	65.0	3329319
37	66.0	2763740
38	67.0	3300911
39	68.0	2395627
40	69.0	2080205
41	70.0	1658179
42	71.0	2315216
43	72.0	1746417
44	73.0	866014
45	74.0	1360049
46	75.0	919980
47	76.0	530976
48	77.0	725843
49	78.0	663327
50	79.0	420769
51	80.0	166450
52	81.0	71752
53	82.0	200676
54	83.0	48281

55	84.0	48122
56	85.0	49224
57	86.0	34010
58	90.0	21514
59	92.0	4182

Pasos siguientes: Generar código con `minutes_by_age` Ver gráficos recomendados

```
# 3
real_trips = trips_df[trips_df['duration'] > 1]
real_trips_count = real_trips.groupby('bike_number')['id'].count()
print('Viajes totales reales:', real_trips_count.sum())
print('Bicicletas totales registradas:', real_trips_count.count())
```

```
Viajes totales reales: 1562109
Bicicletas totales registradas: 1163
```

```
#4
bike_trip_counts = trips_df.groupby('bike_number')['id'].count()

obsolescence_df = bike_trip_counts.to_frame(name='num_trips')
obsolescence_df['obsolescence'] = obsolescence_df['num_trips'] / 1800
obsolescence_df = obsolescence_df.drop(columns=['num_trips']).reset_index()
obsolescence_df = obsolescence_df.sort_values(by='obsolescence')
obsolescence_df
# obsolescence_df['obsolescence'].describe()
```

	bike_number	obsolescence
777	T01064	0.005556
554	B00542	0.013889
152	B00137	0.015556
232	B00218	0.018333
841	T01129	0.027222
...	...	...
575	B00563	1.146667
571	B00559	1.156111
560	B00548	1.164444
281	B00268	1.165556
503	B00490	1.175556

1163 rows × 2 columns

Pasos siguientes: Generar código con `obsolescence_df` Ver gráficos recomendados

```
#5
bikes_trip_count = trips_df.groupby('bike_number')['id'].count().reset_index()
bikes_trip_count.columns = ['bike_number', 'num_trips']
bikes_trip_count
```

	bike_number	num_trips
0	A07799	223
1	A07800	247
2	A07807	325
3	A07808	278
4	A07809	267
...	...	...
1158	T01448	1261
1159	T01449	1344
1160	T01450	1297
1161	T01460	844
1162	b00225	1545

1163 rows × 2 columns

Pasos siguientes: Generar código con `bikes_trip_count` Ver gráficos recomendados

```
#6
null_trips = trips_df[trips_df['duration'] < 60]

null_trip_counts = null_trips.groupby('bike_number')['id'].count().reset_index()
null_trip_counts.columns = ['bike_number', 'num_null_trips']

top_10_null_trips = null_trip_counts.sort_values(by='num_null_trips', ascending=False).head(10)
top_10_null_trips
```

	bike_number	num_null_trips
162	B00156	27
722	T01054	26
333	B00330	22
523	B00523	22
272	B00269	19
561	B00562	19
463	B00460	19
496	B00495	18
363	B00360	18
83	B00075	18

Pasos siguientes: Generar código con `top_10_null_trips` Ver gráficos recomendados

```
#7

bike_age_usage = trips_df[['id','birth_date', 'duration', 'bike_number']]

bike_age_usage = bike_age_usage.dropna()
bike_age_usage = bike_age_usage[bike_age_usage['birth_date'] > 0.0]
bike_age_usage['age'] = current_year - bike_age_usage['birth_date']
bike_age_usage = bike_age_usage.drop(columns=['birth_date'])

bike_age_usage = bike_age_usage.groupby(['age', 'bike_number'])['id'].count().reset_index()
bike_age_usage.columns = ['age', 'bike_number', 'num_trips']

most_used_bike_per_age = bike_age_usage.loc[bike_age_usage.groupby('age')['num_trips'].idxmax()]
most_used_bike_per_age
```

23023	57.0	B00402	27
23539	58.0	B00075	24
24409	59.0	B00120	21
25674	60.0	B00583	16
26463	61.0	B00562	21
27010	62.0	B00285	22
27671	63.0	B00123	18
28389	64.0	B00018	16
29587	65.0	B00407	17
30105	66.0	B00114	18
30820	67.0	B00023	17
32062	68.0	B00490	14
32808	69.0	B00471	12
33306	70.0	B00202	10
33954	71.0	B00127	14
34879	72.0	B00292	13
35374	73.0	B00061	10
36346	74.0	B00471	9
36954	75.0	B00471	8
37364	76.0	B00360	7
37736	77.0	B00195	8
38175	78.0	B00070	5
38858	79.0	B00531	5
38944	80.0	B00006	2
39062	81.0	B00032	2
39185	82.0	B00064	3
39352	83.0	B00074	2

39371	84.0	B00078	2
39441	85.0	B00054	2
39534	86.0	B00590	3
39537	90.0	B00240	1
39543	92.0	B00068	1

Pasos siguientes: Generar código con most_used_bike_per_age | Ver gráficos recomendados

```
#8
trips_ge3 = trips_df[(trips_df['duration'] >= 3)]

trips_ge3_counts = trips_ge3.groupby('bike_number')['id'].count()

bikes_over_2000_trips = trips_ge3_counts[trips_ge3_counts > 2000].reset_index()
bikes_over_2000_trips.columns = ['bike_number', 'num_trips']

bikes_over_2000_trips
```

	bike_number	num_trips
0	B00118	2013
1	B00268	2092
2	B00329	2010
3	B00372	2017
4	B00391	2040
5	B00444	2014
6	B00490	2114
7	B00548	2093
8	B00559	2076
9	B00563	2060