pd

## Librería Pandas para preparación de los datos[¶](#Librería-Pandas-para-preparación-de-los-datos)

Python tiene un montón de librerías para hacer casi todo lo que queramos.

Una gran parte de las utilidades viene por defecto con el propio lenguaje, y otras han sido desarrolladas por la comunidad Python y es necesario instalarlas e importarlas para poder usarlas.

Una de estas librerías es Pandas.

In [1]:

import pandas as pd  
import numpy as np  
import json  
import os  
from os import listdir  
from os.path import isfile, join  
  
import zipfile  
import time  
import re  
from dateutil.parser import \*

Se trata de la librería básica para trabajar con series y con datos en forma de tabla. Con ella es posible importar gran cantidad de datos, ejecutar estadísticas, filtrar los datos, agrupar los datos, y casi todo…

## Análisis de datos de tráfico - aire (C.Madrid)[¶](#Análisis-de-datos-de-tráfico---aire-(C.Madrid))

El portal de datos abiertos del Ayuntamiento de Madrid (<http://datos.madrid.es/portal/site/egob>) pone a disposión gran cantidad de datasets.

* **Datos de observaciones de puntos de medida de tráfico**

La descripción de los datos y los ficheros en formato CSV comprimido se pueden encuentran en los siguientes enlaces:

* [Datos históricos de Tráfico](http://datos.madrid.es/portal/site/egob/menuitem.c05c1f754a33a9fbe4b2e4b284f1a5a0/?vgnextoid=33cb30c367e78410VgnVCM1000000b205a0aRCRD&amp;vgnextchannel=374512b9ace9f310VgnVCM100000171f5a0aRCRD&amp;vgnextfmt=default)

Distintos sensores de control de tráfico de la Comunidad de Madrid proporcionan datos sobre la intensidad de tráfico cada 15 minutos.

Nos planteamos algunas cuestiones:

1. La intensidad de tráfico informa acerca de la movilidad de la población
2. Queremos estudiar una estación de tráfico concreta, conocer la intensidad de tráfico que registra dicha estación a lo largo del més, qué día registra la mayor intensidad, etc.
3. En cuanto a los protocolos de cortes de tráfico impuestos por el ayuntamiento de Madrid, ¿tienen éxito?
4. ¿Existe correlación entre los valores de calidad de aire y la intensidad de tráfico?

## Leer los datos[¶](#Leer-los-datos)

Mostramos cómo importar los datos en Pandas y en un notebook de Jupyter para comenzar con el análisis.

|  |  |
| --- | --- |
| Dataset (Portal del datos abiertos) | Descripción |
| 02-2017\_trafico.zip | Datos de intensidad de tráfico cada 15 minutos |
| 02-2017\_aire.csv | Datos de calidad de aire cada hora |

|  |  |
| --- | --- |
| Otros datasets | Descripción |
| pmed\_trafico\_coord.csv | Geolocalización de las estaciones de tráfico |
| descripcion\_est\_traf.xlsx | Descripción de las estaciones de tráfico |

* **Formato CSV**: registros con datos separados por comas. Es uno de los formatos más habituales.
* **Formato ZIP**: fichero CSV comprimido.
* **Formato Excel**: conocido por todos.

### Crear un dataframe de Pandas[¶](#Crear-un-dataframe-de-Pandas)

La librería Pandas proporciona funciones de lectura y escritura de datos en muchos formatos (txt, csv, xlsx, json, …).

Para leer datos en formato **CSV** utilizamos la función read\_csv de Pandas.

In [2]:

# acceso en modo lectura al fichero comprimido  
zf = zipfile.ZipFile( "./pd\_trafico/02-2017.zip", 'r')  
  
# crear un dataframe llamado tf  
tf = pd.read\_csv(zf.open(zf.namelist()[0]), sep = ';', parse\_dates =[1])

Como se puede observar a continuación, tenemos una estructura de tabla (con filas y columnas) similar a una hoja de cálculo. Una diferncia es que todo lo que hagamos con los datos se podrá reproducir en cualquier momento.

In [3]:

tf

Out[3]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | identif | tipo\_elem | intensidad | ocupacion | carga | vmed | error | periodo\_integracion |
| 0 | 3753 | 2017-02-18 01:45:00 | 09004 | PUNTOS MEDIDA URBANOS | 1986 | 23 | 93 | 0 | N | 15 |
| 1 | 4021 | 2017-02-18 01:45:00 | 09005 | PUNTOS MEDIDA URBANOS | 330 | 1 | 19 | 0 | N | 15 |
| 2 | 4022 | 2017-02-18 01:45:00 | 09006 | PUNTOS MEDIDA URBANOS | 298 | 1 | 31 | 0 | N | 15 |
| 3 | 4023 | 2017-02-18 01:45:00 | 09007 | PUNTOS MEDIDA URBANOS | 263 | 4 | 23 | 0 | N | 15 |
| 4 | 4024 | 2017-02-18 01:45:00 | 09008 | PUNTOS MEDIDA URBANOS | 95 | 0 | 10 | 0 | N | 15 |
| 5 | 4025 | 2017-02-18 01:45:00 | 09009 | PUNTOS MEDIDA URBANOS | 866 | 4 | 21 | 0 | N | 15 |
| 6 | 4026 | 2017-02-18 01:45:00 | 09010 | PUNTOS MEDIDA URBANOS | 422 | 2 | 21 | 0 | N | 15 |
| 7 | 4027 | 2017-02-18 01:45:00 | 09011 | PUNTOS MEDIDA URBANOS | 300 | 3 | 24 | 0 | N | 15 |
| 8 | 4028 | 2017-02-18 01:45:00 | 09012 | PUNTOS MEDIDA URBANOS | 428 | 3 | 22 | 0 | N | 15 |
| 9 | 4029 | 2017-02-18 01:45:00 | 09013 | PUNTOS MEDIDA URBANOS | 334 | 2 | 18 | 0 | N | 15 |
| 10 | 4030 | 2017-02-18 01:45:00 | 09014 | PUNTOS MEDIDA URBANOS | 39 | 2 | 14 | 0 | N | 15 |
| 11 | 4031 | 2017-02-18 01:45:00 | 09015 | PUNTOS MEDIDA URBANOS | 429 | 2 | 19 | 0 | N | 15 |
| 12 | 4032 | 2017-02-18 01:45:00 | 09016 | PUNTOS MEDIDA URBANOS | 538 | 4 | 23 | 0 | N | 15 |
| 13 | 4033 | 2017-02-18 01:45:00 | 09017 | PUNTOS MEDIDA URBANOS | 159 | 2 | 12 | 0 | N | 15 |
| 14 | 4034 | 2017-02-18 01:45:00 | 09018 | PUNTOS MEDIDA URBANOS | 370 | 2 | 26 | 0 | N | 15 |
| 15 | 4035 | 2017-02-18 01:45:00 | 09019 | PUNTOS MEDIDA URBANOS | 255 | 1 | 13 | 0 | N | 15 |
| 16 | 3614 | 2017-02-18 01:45:00 | 11009 | PUNTOS MEDIDA URBANOS | 124 | 3 | 7 | 0 | N | 15 |
| 17 | 3615 | 2017-02-18 01:45:00 | 11010 | PUNTOS MEDIDA URBANOS | 86 | 1 | 2 | 0 | N | 15 |
| 18 | 4084 | 2017-02-18 01:45:00 | 11011 | PUNTOS MEDIDA URBANOS | 20 | 0 | 4 | 0 | N | 15 |
| 19 | 4085 | 2017-02-18 01:45:00 | 11012 | PUNTOS MEDIDA URBANOS | 0 | 0 | 0 | 0 | N | 15 |
| 20 | 5095 | 2017-02-18 01:45:00 | 11013 | PUNTOS MEDIDA URBANOS | 14 | 0 | 6 | 0 | N | 15 |
| 21 | 5096 | 2017-02-18 01:45:00 | 11014 | PUNTOS MEDIDA URBANOS | 56 | 0 | 13 | 0 | N | 15 |
| 22 | 9841 | 2017-02-05 19:30:00 | 03001 | PUNTOS MEDIDA URBANOS | 507 | 1 | 12 | 0 | N | 15 |
| 23 | 4086 | 2017-02-18 01:45:00 | 11015 | PUNTOS MEDIDA URBANOS | 158 | 1 | 15 | 0 | N | 15 |
| 24 | 4088 | 2017-02-18 01:45:00 | 11017 | PUNTOS MEDIDA URBANOS | 36 | 1 | 11 | 0 | N | 15 |
| 25 | 3928 | 2017-02-05 19:30:00 | 04022 | PUNTOS MEDIDA URBANOS | 17 | 20 | 13 | 0 | N | 7 |
| 26 | 4089 | 2017-02-18 01:45:00 | 11018 | PUNTOS MEDIDA URBANOS | 5 | 0 | 4 | 0 | N | 15 |
| 27 | 4090 | 2017-02-18 01:45:00 | 11019 | PUNTOS MEDIDA URBANOS | 199 | 1 | 16 | 0 | N | 15 |
| 28 | 4091 | 2017-02-18 01:45:00 | 11020 | PUNTOS MEDIDA URBANOS | 114 | 1 | 13 | 0 | N | 15 |
| 29 | 4092 | 2017-02-18 01:45:00 | 11021 | PUNTOS MEDIDA URBANOS | 50 | 0 | 8 | 0 | N | 15 |
| … | … | … | … | … | … | … | … | … | … | … |
| 8823487 | 3819 | 2017-02-28 09:45:00 | PM10502 | PUNTOS MEDIDA M-30 | 6829 | 11 | 60 | 76 | N | 15 |
| 8823488 | 3820 | 2017-02-28 09:45:00 | PM10561 | PUNTOS MEDIDA M-30 | 5128 | 15 | 73 | 83 | N | 15 |
| 8823489 | 6663 | 2017-02-28 09:45:00 | PM10562 | PUNTOS MEDIDA M-30 | 4300 | 14 | 50 | 68 | N | 15 |
| 8823490 | 3799 | 2017-02-28 09:45:00 | PM10611 | PUNTOS MEDIDA M-30 | 4940 | 10 | 53 | 89 | N | 15 |
| 8823491 | 3800 | 2017-02-28 09:45:00 | PM10612 | PUNTOS MEDIDA M-30 | 2312 | 9 | 51 | 80 | N | 15 |
| 8823492 | 6664 | 2017-02-28 09:45:00 | PM10661 | PUNTOS MEDIDA M-30 | 4008 | 11 | 59 | 88 | N | 15 |
| 8823493 | 6665 | 2017-02-28 09:45:00 | PM10662 | PUNTOS MEDIDA M-30 | 3724 | 9 | 44 | 76 | N | 15 |
| 8823494 | 4225 | 2017-02-28 09:45:00 | 06009 | PUNTOS MEDIDA URBANOS | 408 | 2 | 49 | 0 | N | 5 |
| 8823495 | 4214 | 2017-02-28 09:45:00 | 14018 | PUNTOS MEDIDA URBANOS | 590 | 24 | 69 | 0 | N | 4 |
| 8823496 | 4387 | 2017-02-28 09:45:00 | 18010 | PUNTOS MEDIDA URBANOS | 430 | 7 | 23 | 0 | N | 6 |
| 8823497 | 6926 | 2017-02-28 09:45:00 | 55063 | PUNTOS MEDIDA URBANOS | 318 | 3 | 29 | 0 | N | 9 |
| 8823498 | 6286 | 2017-02-28 09:45:00 | 56001 | PUNTOS MEDIDA URBANOS | 387 | 45 | 38 | 0 | N | 14 |
| 8823499 | 6520 | 2017-02-28 09:45:00 | 71020 | PUNTOS MEDIDA URBANOS | 362 | 4 | 17 | 0 | N | 13 |
| 8823500 | 6628 | 2017-02-28 09:45:00 | 79403 | PUNTOS MEDIDA URBANOS | 26 | 1 | 13 | 0 | N | 7 |
| 8823501 | 5199 | 2017-02-28 09:45:00 | 82001 | PUNTOS MEDIDA URBANOS | 530 | 9 | 79 | 0 | N | 14 |
| 8823502 | 9834 | 2017-02-28 09:45:00 | 83007 | PUNTOS MEDIDA URBANOS | 891 | 8 | 83 | 0 | N | 14 |
| 8823503 | 5170 | 2017-02-28 09:45:00 | 80027 | PUNTOS MEDIDA URBANOS | 103 | 0 | 7 | 0 | N | 14 |
| 8823504 | 4590 | 2017-02-28 09:45:00 | 33006 | PUNTOS MEDIDA URBANOS | 36 | 1 | 5 | 0 | N | 10 |
| 8823505 | 4591 | 2017-02-28 09:45:00 | 33007 | PUNTOS MEDIDA URBANOS | 36 | 1 | 5 | 0 | N | 10 |
| 8823506 | 5449 | 2017-02-28 09:45:00 | 43038 | PUNTOS MEDIDA URBANOS | 42 | 5 | 7 | 0 | N | 13 |
| 8823507 | 4726 | 2017-02-28 09:45:00 | 93009 | PUNTOS MEDIDA URBANOS | 43 | 5 | 7 | 0 | N | 14 |
| 8823508 | 9911 | 2017-02-28 09:45:00 | 93032 | PUNTOS MEDIDA URBANOS | 99 | 2 | 8 | 0 | N | 15 |
| 8823509 | 9910 | 2017-02-28 09:45:00 | 93033 | PUNTOS MEDIDA URBANOS | 59 | 2 | 4 | 0 | N | 15 |
| 8823510 | 9898 | 2017-02-28 09:45:00 | 93034 | PUNTOS MEDIDA URBANOS | 1129 | 14 | 46 | 0 | N | 15 |
| 8823511 | 9899 | 2017-02-28 09:45:00 | 93035 | PUNTOS MEDIDA URBANOS | 212 | 2 | 9 | 0 | N | 15 |
| 8823512 | 9900 | 2017-02-28 09:45:00 | 93036 | PUNTOS MEDIDA URBANOS | 463 | 3 | 17 | 0 | N | 15 |
| 8823513 | 9975 | 2017-02-28 09:45:00 | 93037 | PUNTOS MEDIDA URBANOS | 292 | 3 | 12 | 0 | N | 15 |
| 8823514 | 9974 | 2017-02-28 09:45:00 | 93038 | PUNTOS MEDIDA URBANOS | 488 | 4 | 18 | 0 | N | 15 |
| 8823515 | 9973 | 2017-02-28 09:45:00 | 93039 | PUNTOS MEDIDA URBANOS | 83 | 2 | 6 | 0 | N | 15 |
| 8823516 | 9976 | 2017-02-28 09:45:00 | 93040 | PUNTOS MEDIDA URBANOS | 175 | 3 | 12 | 0 | N | 15 |

8823517 rows × 10 columns

Los dataframes de Pandas tienen asociadas un montón de operaciones o metodos:

**El método head()**

In [4]:

tf.head()

Out[4]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | identif | tipo\_elem | intensidad | ocupacion | carga | vmed | error | periodo\_integracion |
| 0 | 3753 | 2017-02-18 01:45:00 | 09004 | PUNTOS MEDIDA URBANOS | 1986 | 23 | 93 | 0 | N | 15 |
| 1 | 4021 | 2017-02-18 01:45:00 | 09005 | PUNTOS MEDIDA URBANOS | 330 | 1 | 19 | 0 | N | 15 |
| 2 | 4022 | 2017-02-18 01:45:00 | 09006 | PUNTOS MEDIDA URBANOS | 298 | 1 | 31 | 0 | N | 15 |
| 3 | 4023 | 2017-02-18 01:45:00 | 09007 | PUNTOS MEDIDA URBANOS | 263 | 4 | 23 | 0 | N | 15 |
| 4 | 4024 | 2017-02-18 01:45:00 | 09008 | PUNTOS MEDIDA URBANOS | 95 | 0 | 10 | 0 | N | 15 |

**El método info()**

In [5]:

tf.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 8823517 entries, 0 to 8823516  
Data columns (total 10 columns):  
idelem int64  
fecha datetime64[ns]  
identif object  
tipo\_elem object  
intensidad int64  
ocupacion int64  
carga int64  
vmed int64  
error object  
periodo\_integracion int64  
dtypes: datetime64[ns](1), int64(6), object(3)  
memory usage: 673.2+ MB

#### Descripción de los datos de tráfico[¶](#Descripción-de-los-datos-de-tráfico)

En el propio portal podemos encontrar información acerca del contenido de los ficheros ([info tráfico](http://datos.madrid.es/FWProjects/egob/contenidos/datasets/ficheros/Estructura%20y%20contenido%20del%20fichero%20csv.pdf)).

* idelem: Identificación única del Punto de Medida en los sistemas de control del tráfico del Ayuntamiento de Madrid.
* fecha: fecha en formato Año, mes, dia, hora, minutos, segundos
* intensidad: Intensidad del Punto de Medida en el periodo de 15 minutos (vehículos/hora).
* vmed: Velocidad media de los vehículos en el periodo de 15 minutos (Km./h). Sólo para puntos de medida interurbanos M30.
* error: Indicación de si ha habido al menos una muestra errónea o sustituida en el periodo de 15 minutos.
  + N: no ha habido errores ni sustituciones
  + E: los parámetros de calidad de alguna de las muestras integradas no son óptimos.
* periodo\_integracion: Número de muestras recibidas y consideradas

En realidad, no necesitamos todas columnas para nuestro análisis. Cargamos de nuevo el dataset, pero esta vez con menos columnas.

In [6]:

# crear un dataframe llamado tf  
tf = pd.read\_csv(zf.open(zf.namelist()[0]), sep = ';',   
 usecols = [0,1,3,4,5,9], # leemos solo 6 columnas  
 parse\_dates =[1])

In [7]:

tf.head()

Out[7]:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | tipo\_elem | intensidad | ocupacion | periodo\_integracion |
| 0 | 3753 | 2017-02-18 01:45:00 | PUNTOS MEDIDA URBANOS | 1986 | 23 | 15 |
| 1 | 4021 | 2017-02-18 01:45:00 | PUNTOS MEDIDA URBANOS | 330 | 1 | 15 |
| 2 | 4022 | 2017-02-18 01:45:00 | PUNTOS MEDIDA URBANOS | 298 | 1 | 15 |
| 3 | 4023 | 2017-02-18 01:45:00 | PUNTOS MEDIDA URBANOS | 263 | 4 | 15 |
| 4 | 4024 | 2017-02-18 01:45:00 | PUNTOS MEDIDA URBANOS | 95 | 0 | 15 |

### Acceso a columnas[¶](#Acceso-a-columnas)

Para ver los datos de una columna separada del resto de los datos es necesario escribir el nombre del dataframe, seguido del nombre de la columna.

In [8]:

# Accesso al tipo de elemento  
tf.tipo\_elem

Out[8]:

0 PUNTOS MEDIDA URBANOS  
1 PUNTOS MEDIDA URBANOS  
2 PUNTOS MEDIDA URBANOS  
3 PUNTOS MEDIDA URBANOS  
4 PUNTOS MEDIDA URBANOS  
5 PUNTOS MEDIDA URBANOS  
6 PUNTOS MEDIDA URBANOS  
7 PUNTOS MEDIDA URBANOS  
8 PUNTOS MEDIDA URBANOS  
9 PUNTOS MEDIDA URBANOS  
10 PUNTOS MEDIDA URBANOS  
11 PUNTOS MEDIDA URBANOS  
12 PUNTOS MEDIDA URBANOS  
13 PUNTOS MEDIDA URBANOS  
14 PUNTOS MEDIDA URBANOS  
15 PUNTOS MEDIDA URBANOS  
16 PUNTOS MEDIDA URBANOS  
17 PUNTOS MEDIDA URBANOS  
18 PUNTOS MEDIDA URBANOS  
19 PUNTOS MEDIDA URBANOS  
20 PUNTOS MEDIDA URBANOS  
21 PUNTOS MEDIDA URBANOS  
22 PUNTOS MEDIDA URBANOS  
23 PUNTOS MEDIDA URBANOS  
24 PUNTOS MEDIDA URBANOS  
25 PUNTOS MEDIDA URBANOS  
26 PUNTOS MEDIDA URBANOS  
27 PUNTOS MEDIDA URBANOS  
28 PUNTOS MEDIDA URBANOS  
29 PUNTOS MEDIDA URBANOS  
 ...   
8823487 PUNTOS MEDIDA M-30  
8823488 PUNTOS MEDIDA M-30  
8823489 PUNTOS MEDIDA M-30  
8823490 PUNTOS MEDIDA M-30  
8823491 PUNTOS MEDIDA M-30  
8823492 PUNTOS MEDIDA M-30  
8823493 PUNTOS MEDIDA M-30  
8823494 PUNTOS MEDIDA URBANOS  
8823495 PUNTOS MEDIDA URBANOS  
8823496 PUNTOS MEDIDA URBANOS  
8823497 PUNTOS MEDIDA URBANOS  
8823498 PUNTOS MEDIDA URBANOS  
8823499 PUNTOS MEDIDA URBANOS  
8823500 PUNTOS MEDIDA URBANOS  
8823501 PUNTOS MEDIDA URBANOS  
8823502 PUNTOS MEDIDA URBANOS  
8823503 PUNTOS MEDIDA URBANOS  
8823504 PUNTOS MEDIDA URBANOS  
8823505 PUNTOS MEDIDA URBANOS  
8823506 PUNTOS MEDIDA URBANOS  
8823507 PUNTOS MEDIDA URBANOS  
8823508 PUNTOS MEDIDA URBANOS  
8823509 PUNTOS MEDIDA URBANOS  
8823510 PUNTOS MEDIDA URBANOS  
8823511 PUNTOS MEDIDA URBANOS  
8823512 PUNTOS MEDIDA URBANOS  
8823513 PUNTOS MEDIDA URBANOS  
8823514 PUNTOS MEDIDA URBANOS  
8823515 PUNTOS MEDIDA URBANOS  
8823516 PUNTOS MEDIDA URBANOS  
Name: tipo\_elem, dtype: object

Cada columna es un elemento de Pandas denominado Serie.

**Contar el número de valores distintos de una columna**

Las series de Pandas también tienen un gran conjunto de operaciones y métodos. El mátodo value\_counts() permite calcular el número de valores distintos que aparece en una columna.

In [9]:

tf.tipo\_elem.value\_counts()

Out[9]:

PUNTOS MEDIDA URBANOS 7816294  
PUNTOS MEDIDA M-30 1007223  
Name: tipo\_elem, dtype: int64

El valor devuelto por el método value\_counts() es una serie. Para convertirlo en dataframe utilizamos el método **reset\_index()**.

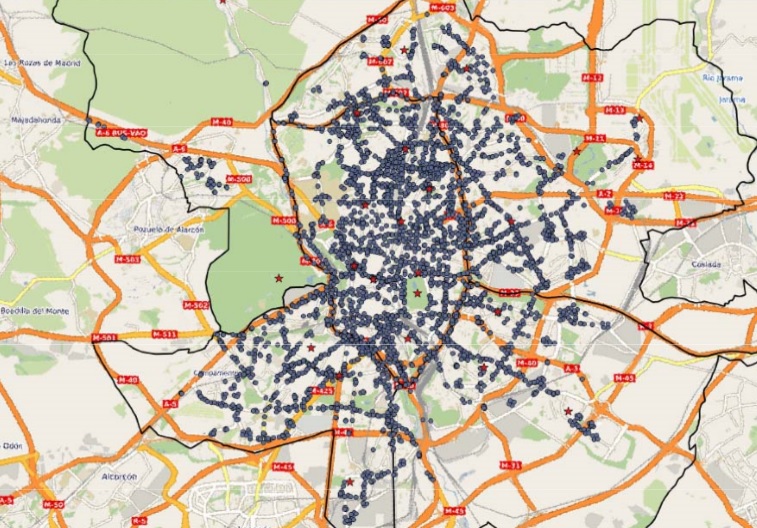
In [10]:

tf.tipo\_elem.value\_counts().reset\_index()

Out[10]:

|  |  |  |
| --- | --- | --- |
|  | index | tipo\_elem |
| 0 | PUNTOS MEDIDA URBANOS | 7816294 |
| 1 | PUNTOS MEDIDA M-30 | 1007223 |

La siguiente figura muestra en un mapa las distintas estaciones de tráfico y las estaciones de aire.



Podemos preguntar acerca del número de muestras consideradas para el cálculo de la intensidad:

In [11]:

tf.periodo\_integracion.value\_counts().reset\_index()

Out[11]:

|  |  |  |
| --- | --- | --- |
|  | index | periodo\_integracion |
| 0 | 15 | 6738368 |
| 1 | 14 | 489858 |
| 2 | 5 | 373479 |
| 3 | 13 | 253298 |
| 4 | 12 | 161149 |
| 5 | 11 | 113305 |
| 6 | 1 | 113039 |
| 7 | 10 | 91408 |
| 8 | 3 | 84858 |
| 9 | 2 | 84365 |
| 10 | 9 | 73266 |
| 11 | 8 | 66651 |
| 12 | 4 | 66297 |
| 13 | 7 | 57696 |
| 14 | 6 | 56479 |
| 15 | 16 | 1 |

#### Para visualización[¶](#Para-visualización)

In [12]:

import seaborn as sns  
import matplotlib.pyplot as plt  
%matplotlib inline  
  
sns.set(style="white", context="talk")  
  
ranking = tf.periodo\_integracion.value\_counts().reset\_index().sort\_values(by = 'index')  
  
# figuras  
f, ax1 = plt.subplots(1, figsize=(8, 6))  
  
# Ejemplo 1  
sns.barplot(ranking['index'], ranking.periodo\_integracion, palette="BuGn\_d", ax=ax1)  
ax1.set\_ylabel("Numero de observaciones");

![](data:image/png;base64;base64,)

### Análisis estadístico[¶](#Análisis-estadístico)

Con los dataframes creados es posible realizar operaciones estadísticas mediante las funciones y métodos proporcionados por la librería. Por ejemplo, la media, la suma, etc…

In [13]:

# valor máximo de intensidad en el mes de febrero de 2017  
tf.intensidad.max()

Out[13]:

99999

In [14]:

# valor mínimo de intensidad en el mes de febrero de 2017  
tf.intensidad.min()

Out[14]:

0

In [15]:

# valor medio  
tf.intensidad.mean()

Out[15]:

491.7426563580033

In [16]:

# desviación típica  
tf.intensidad.std()

Out[16]:

760.7516804691924

In [17]:

# suma de todas la intensidades  
tf.intensidad.sum()

Out[17]:

43932392

El método describe() calcula los primeros estadísticos para las columnas numéricas:

In [18]:

tf.describe()

Out[18]:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | idelem | intensidad | ocupacion | periodo\_integracion |
| count | 8.823517e+06 | 8.823517e+06 | 8.823517e+06 | 8.823517e+06 |
| mean | 5.500865e+03 | 4.917427e+02 | 6.736623e+00 | 1.359151e+01 |
| std | 1.667975e+03 | 7.607517e+02 | 1.108778e+01 | 3.327914e+00 |
| min | 1.001000e+03 | 0.000000e+00 | -1.000000e+00 | 1.000000e+00 |
| 25% | 4.275000e+03 | 7.400000e+01 | 1.000000e+00 | 1.500000e+01 |
| 50% | 5.452000e+03 | 2.300000e+02 | 3.000000e+00 | 1.500000e+01 |
| 75% | 6.373000e+03 | 5.890000e+02 | 8.000000e+00 | 1.500000e+01 |
| max | 1.018100e+04 | 9.999900e+04 | 1.000000e+02 | 1.600000e+01 |

## Transformar datos[¶](#Transformar-datos)

Pandas dispone de gran cantidad de funciones para el tratamiento de datos de tipo texto.

Una transformación muy sencilla es la de acortar los nombre de la columna tipo\_elem. Por ejemplo eliminado el prefijo PUNTOS MEDIDA. Para ello utilizamos el método apply junto con la función replace.

In [19]:

tf.tipo\_elem = tf.tipo\_elem.apply(lambda x :x.replace('PUNTOS MEDIDA ', ''))

In [20]:

tf.head()

Out[20]:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | tipo\_elem | intensidad | ocupacion | periodo\_integracion |
| 0 | 3753 | 2017-02-18 01:45:00 | URBANOS | 1986 | 23 | 15 |
| 1 | 4021 | 2017-02-18 01:45:00 | URBANOS | 330 | 1 | 15 |
| 2 | 4022 | 2017-02-18 01:45:00 | URBANOS | 298 | 1 | 15 |
| 3 | 4023 | 2017-02-18 01:45:00 | URBANOS | 263 | 4 | 15 |
| 4 | 4024 | 2017-02-18 01:45:00 | URBANOS | 95 | 0 | 15 |

In [21]:

tf.tipo\_elem.value\_counts()

Out[21]:

URBANOS 7816294  
M-30 1007223  
Name: tipo\_elem, dtype: int64

In [22]:

tf.tipo\_elem = tf.tipo\_elem.apply(lambda x :x.title())

In [23]:

tf.head()

Out[23]:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | tipo\_elem | intensidad | ocupacion | periodo\_integracion |
| 0 | 3753 | 2017-02-18 01:45:00 | Urbanos | 1986 | 23 | 15 |
| 1 | 4021 | 2017-02-18 01:45:00 | Urbanos | 330 | 1 | 15 |
| 2 | 4022 | 2017-02-18 01:45:00 | Urbanos | 298 | 1 | 15 |
| 3 | 4023 | 2017-02-18 01:45:00 | Urbanos | 263 | 4 | 15 |
| 4 | 4024 | 2017-02-18 01:45:00 | Urbanos | 95 | 0 | 15 |

In [24]:

tf.tipo\_elem.value\_counts()

Out[24]:

Urbanos 7816294  
M-30 1007223  
Name: tipo\_elem, dtype: int64

In [25]:

tf.corr()

Out[25]:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | idelem | intensidad | ocupacion | periodo\_integracion |
| idelem | 1.000000 | -0.002802 | -0.019175 | 0.033739 |
| intensidad | -0.002802 | 1.000000 | 0.200032 | 0.039879 |
| ocupacion | -0.019175 | 0.200032 | 1.000000 | 0.055740 |
| periodo\_integracion | 0.033739 | 0.039879 | 0.055740 | 1.000000 |

## Filtros[¶](#Filtros)

Supongamos que queremos centrar el análisis en las estaciones de tráfico del interior a la M30.

En Pandas, para seleccionar un conjunto de tuplas utilizamos lo que se denominan filtros, que no son más que una serie con valores que indican si la tupla debe ser filtrada o no. Los filtros se crean utilizado operadores de comparación (==, >, <, …).

In [26]:

tf.tipo\_elem == 'M-30'

Out[26]:

0 False  
1 False  
2 False  
3 False  
4 False  
5 False  
6 False  
7 False  
8 False  
9 False  
10 False  
11 False  
12 False  
13 False  
14 False  
15 False  
16 False  
17 False  
18 False  
19 False  
20 False  
21 False  
22 False  
23 False  
24 False  
25 False  
26 False  
27 False  
28 False  
29 False  
 ...   
8823487 True  
8823488 True  
8823489 True  
8823490 True  
8823491 True  
8823492 True  
8823493 True  
8823494 False  
8823495 False  
8823496 False  
8823497 False  
8823498 False  
8823499 False  
8823500 False  
8823501 False  
8823502 False  
8823503 False  
8823504 False  
8823505 False  
8823506 False  
8823507 False  
8823508 False  
8823509 False  
8823510 False  
8823511 False  
8823512 False  
8823513 False  
8823514 False  
8823515 False  
8823516 False  
Name: tipo\_elem, dtype: bool

In [27]:

tf\_m30 = tf[tf.tipo\_elem == 'M-30']

In [28]:

tf\_m30.head()

Out[28]:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | tipo\_elem | intensidad | ocupacion | periodo\_integracion |
| 315 | 1010 | 2017-02-13 11:45:00 | M-30 | 2640 | 5 | 5 |
| 317 | 1044 | 2017-02-13 11:45:00 | M-30 | 948 | 6 | 5 |
| 330 | 3542 | 2017-02-13 11:45:00 | M-30 | 2592 | 7 | 5 |
| 331 | 7119 | 2017-02-13 11:45:00 | M-30 | 1728 | 7 | 5 |
| 402 | 1021 | 2017-02-13 11:45:00 | M-30 | 2364 | 6 | 5 |

In [29]:

tf\_m30.info()

<class 'pandas.core.frame.DataFrame'>  
Int64Index: 1007223 entries, 315 to 8823493  
Data columns (total 6 columns):  
idelem 1007223 non-null int64  
fecha 1007223 non-null datetime64[ns]  
tipo\_elem 1007223 non-null object  
intensidad 1007223 non-null int64  
ocupacion 1007223 non-null int64  
periodo\_integracion 1007223 non-null int64  
dtypes: datetime64[ns](1), int64(4), object(1)  
memory usage: 53.8+ MB

Seleccionar solo aquellas tuplas o filas de las estaaciones situadas dentro de la M-30 con intensidad de tráfico superior a 5000.

In [30]:

# intensidad > 1000  
alta = tf\_m30[tf\_m30.intensidad > 5000]

In [31]:

alta.tail()

Out[31]:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | tipo\_elem | intensidad | ocupacion | periodo\_integracion |
| 8823355 | 6840 | 2017-02-28 09:45:00 | M-30 | 6456 | 21 | 5 |
| 8823356 | 6839 | 2017-02-28 09:45:00 | M-30 | 5256 | 19 | 5 |
| 8823359 | 6843 | 2017-02-28 09:45:00 | M-30 | 5376 | 11 | 5 |
| 8823487 | 3819 | 2017-02-28 09:45:00 | M-30 | 6829 | 11 | 15 |
| 8823488 | 3820 | 2017-02-28 09:45:00 | M-30 | 5128 | 15 | 15 |

In [32]:

alta.info()

<class 'pandas.core.frame.DataFrame'>  
Int64Index: 35846 entries, 2142 to 8823488  
Data columns (total 6 columns):  
idelem 35846 non-null int64  
fecha 35846 non-null datetime64[ns]  
tipo\_elem 35846 non-null object  
intensidad 35846 non-null int64  
ocupacion 35846 non-null int64  
periodo\_integracion 35846 non-null int64  
dtypes: datetime64[ns](1), int64(4), object(1)  
memory usage: 1.9+ MB

## Crear nuevos datos[¶](#Crear-nuevos-datos)

Cuando trabamos con datos, es muy habitual construir nuevos datos a partir de otros. Por ejemplo, puede ser útil añadir a los datos de tráfico una nueva columna que represete el día de la semana.

El método map permite aplicar funciones a cada uno de los valores de una columna del dataframe para producir una nueva columna.

In [33]:

col\_dia = tf\_m30.fecha.map(lambda x : x.day) # crear una nueva columna

In [34]:

col\_dia.value\_counts()

Out[34]:

26 36468  
25 36388  
24 36376  
27 36371  
28 36327  
5 36165  
14 36161  
17 36140  
20 36119  
8 36093  
7 36064  
15 36035  
3 36034  
4 36020  
10 36012  
1 36009  
16 35972  
18 35955  
13 35945  
6 35905  
2 35891  
23 35853  
11 35845  
9 35841  
12 35776  
19 35710  
21 35608  
22 34140  
Name: fecha, dtype: int64

In [35]:

tf\_m30.insert(2, 'dia', col\_dia) # insertarla en el dataframe  
tf\_m30.head()

Out[35]:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | dia | tipo\_elem | intensidad | ocupacion | periodo\_integracion |
| 315 | 1010 | 2017-02-13 11:45:00 | 13 | M-30 | 2640 | 5 | 5 |
| 317 | 1044 | 2017-02-13 11:45:00 | 13 | M-30 | 948 | 6 | 5 |
| 330 | 3542 | 2017-02-13 11:45:00 | 13 | M-30 | 2592 | 7 | 5 |
| 331 | 7119 | 2017-02-13 11:45:00 | 13 | M-30 | 1728 | 7 | 5 |
| 402 | 1021 | 2017-02-13 11:45:00 | 13 | M-30 | 2364 | 6 | 5 |

## Combinación de datos[¶](#Combinación-de-datos)

Para las estaciones de tráfico de la M30, conocemos el identificador, pero no tenemos información geográfica ni una descripción.

Queremos conocer la intensidad de tráfico registrada por la estación P1 Castellana S-N - Pl. Colon-Ayala, el día en que se ha producido mayor intensidad, etc…

La informción de los nombres o descripciones de las estaciones se encuentran en otro dataset. Esta vez se trata deun fichero en forma Excel. Pandas proporciona la función read\_excel para crear un dataframe a partir de una hoja de cálculo.

In [36]:

desc\_tr = pd.read\_excel('./pd\_trafico/descripcion\_est\_traf.xlsx')  
desc\_tr.head()

Out[36]:

|  |  |  |
| --- | --- | --- |
|  | idelem | nombre |
| 0 | 16007 | Cuesta San Vicente O-E - Gran Via-Ferraz |
| 1 | 16005 | San Leonardo E-O - Maestro Guerrero-Princesa |
| 2 | 17021 | (AFOROS) Princesa S-N - San Leonardo-Duque de … |
| 3 | 16042 | Cuesta San Vicente O-E - Bailen-Gran Via |
| 4 | 16301 | (MICRO) Cuesta San Vicente S-N(Delante G.2)(P…. |

In [37]:

coor\_tf = pd.read\_csv('./pd\_trafico/pmed\_trafico\_coord.csv', sep = ';', decimal= b',',  
 usecols = ['idelem', 'Xcoord', 'Ycoord'])  
coor\_tf.head()

Out[37]:

|  |  |  |  |
| --- | --- | --- | --- |
|  | idelem | Xcoord | Ycoord |
| 0 | 1044 | -3.720754 | 40.419125 |
| 1 | 3600 | -3.664409 | 40.406252 |
| 2 | 3705 | -3.709906 | 40.390721 |
| 3 | 6823 | -3.717737 | 40.384554 |
| 4 | 7033 | -3.690105 | 40.426231 |

Como podemos observar, los dataframes recién creados contienen la columna idelem que representa el código de la estación de tráfico.

Lo que haremos a continuación será unir la información de ambos dataframes en un único dataframe. Utilizamos la función merge de Pandas.

In [38]:

merged\_1 = pd.merge(desc\_tr, coor\_tf, on = 'idelem' )

In [39]:

merged\_1.head()

Out[39]:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | idelem | nombre | Xcoord | Ycoord |
| 0 | 1021 | Hortaleza, 75 S-N - Travesia San Mateo-Fernand… | -3.702702 | 40.393459 |
| 1 | 1020 | Fernado VI E-O - Campoamor-Hortaleza | -3.698097 | 40.389509 |
| 2 | 1010 | Pl. Santa Barbara S-N - Serrano Anguita-Pl. A… | -3.725040 | 40.416313 |
| 3 | 1017 | Barbara Braganza O-E - Conde Xiquena-Tamayo y… | -3.725528 | 40.422277 |
| 4 | 1009 | (AFOROS) Genova 13 E-O - Zurbano-Campoamor | -3.724906 | 40.416234 |

In [40]:

merged = pd.merge(tf\_m30, merged\_1, on = 'idelem' )

In [41]:

merged.head()

Out[41]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | dia | tipo\_elem | intensidad | ocupacion | periodo\_integracion | nombre | Xcoord | Ycoord |
| 0 | 1010 | 2017-02-13 11:45:00 | 13 | M-30 | 2640 | 5 | 5 | Pl. Santa Barbara S-N - Serrano Anguita-Pl. A… | -3.72504 | 40.416313 |
| 1 | 1010 | 2017-02-22 15:15:00 | 22 | M-30 | 1320 | 3 | 5 | Pl. Santa Barbara S-N - Serrano Anguita-Pl. A… | -3.72504 | 40.416313 |
| 2 | 1010 | 2017-02-01 21:30:00 | 1 | M-30 | 3036 | 7 | 5 | Pl. Santa Barbara S-N - Serrano Anguita-Pl. A… | -3.72504 | 40.416313 |
| 3 | 1010 | 2017-02-09 16:30:00 | 9 | M-30 | 4680 | 11 | 5 | Pl. Santa Barbara S-N - Serrano Anguita-Pl. A… | -3.72504 | 40.416313 |
| 4 | 1010 | 2017-02-17 07:15:00 | 17 | M-30 | 1168 | 2 | 5 | Pl. Santa Barbara S-N - Serrano Anguita-Pl. A… | -3.72504 | 40.416313 |

Para conocer la intensidad de tráfico registrada por la estación P1 Castellana S-N - Pl. Colon-Ayala tenemos que utilizar los filtros.

In [42]:

tf\_castellana = merged[merged.nombre == 'P1 Castellana S-N - Pl. Colon-Ayala']  
tf\_castellana.head()

Out[42]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | dia | tipo\_elem | intensidad | ocupacion | periodo\_integracion | nombre | Xcoord | Ycoord |
| 28861 | 1030 | 2017-02-11 00:30:00 | 11 | M-30 | 228 | 3 | 5 | P1 Castellana S-N - Pl. Colon-Ayala | -3.706254 | 40.395465 |
| 28862 | 1030 | 2017-02-01 21:30:00 | 1 | M-30 | 576 | 7 | 5 | P1 Castellana S-N - Pl. Colon-Ayala | -3.706254 | 40.395465 |
| 28863 | 1030 | 2017-02-10 08:45:00 | 10 | M-30 | 492 | 7 | 5 | P1 Castellana S-N - Pl. Colon-Ayala | -3.706254 | 40.395465 |
| 28864 | 1030 | 2017-02-03 12:00:00 | 3 | M-30 | 552 | 8 | 5 | P1 Castellana S-N - Pl. Colon-Ayala | -3.706254 | 40.395465 |
| 28865 | 1030 | 2017-02-17 07:15:00 | 17 | M-30 | 408 | 5 | 5 | P1 Castellana S-N - Pl. Colon-Ayala | -3.706254 | 40.395465 |

In [43]:

tf\_castellana.intensidad.max()

Out[43]:

1104

In [44]:

tf\_castellana[tf\_castellana.intensidad == tf\_castellana.intensidad.max()]

Out[44]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | dia | tipo\_elem | intensidad | ocupacion | periodo\_integracion | nombre | Xcoord | Ycoord |
| 29879 | 1030 | 2017-02-10 14:45:00 | 10 | M-30 | 1104 | 15 | 5 | P1 Castellana S-N - Pl. Colon-Ayala | -3.706254 | 40.395465 |

El día 10 de Febrero a las 14:45 se registra un pico en intensidad de tráfico en la estación P1 Castellana S-N - Pl. Colon-Ayala.

## Ordenar datos[¶](#Ordenar-datos)

Otra tarea muy típica cuando se trabaja con datos es la ordenación. Utiizamos el método sort\_values() de Pandas indicando el campo de ordenación.

In [45]:

merged.sort\_values('intensidad', ascending = False)

Out[45]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | dia | tipo\_elem | intensidad | ocupacion | periodo\_integracion | nombre | Xcoord | Ycoord |
| 24137 | 1003 | 2017-02-07 07:45:00 | 7 | M-30 | 7656 | 26 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25277 | 1003 | 2017-02-16 07:45:00 | 16 | M-30 | 7536 | 24 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26165 | 1003 | 2017-02-28 07:45:00 | 28 | M-30 | 7536 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 43522 | 1018 | 2017-02-16 15:30:00 | 16 | M-30 | 7512 | 14 | 5 | (AFOROS) Pa Castellana N-S - Eduardo Dato-Marq… | -3.724722 | 40.421812 |
| 25488 | 1003 | 2017-02-20 08:00:00 | 20 | M-30 | 7500 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23739 | 1003 | 2017-02-01 07:45:00 | 1 | M-30 | 7440 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25978 | 1003 | 2017-02-24 07:30:00 | 24 | M-30 | 7428 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25909 | 1003 | 2017-02-23 07:45:00 | 23 | M-30 | 7416 | 23 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25028 | 1003 | 2017-02-15 08:30:00 | 15 | M-30 | 7404 | 23 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23965 | 1003 | 2017-02-03 08:30:00 | 3 | M-30 | 7392 | 28 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26125 | 1003 | 2017-02-27 07:45:00 | 27 | M-30 | 7392 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25543 | 1003 | 2017-02-20 08:30:00 | 20 | M-30 | 7380 | 24 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 24269 | 1003 | 2017-02-17 07:45:00 | 17 | M-30 | 7380 | 24 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25979 | 1003 | 2017-02-24 07:45:00 | 24 | M-30 | 7380 | 27 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23917 | 1003 | 2017-02-03 07:45:00 | 3 | M-30 | 7344 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25690 | 1003 | 2017-02-21 08:45:00 | 21 | M-30 | 7332 | 23 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23927 | 1003 | 2017-02-02 07:30:00 | 2 | M-30 | 7332 | 24 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 24374 | 1003 | 2017-02-14 08:00:00 | 14 | M-30 | 7320 | 28 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25667 | 1003 | 2017-02-21 07:45:00 | 21 | M-30 | 7320 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25900 | 1003 | 2017-02-23 07:30:00 | 23 | M-30 | 7308 | 24 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25744 | 1003 | 2017-02-22 07:45:00 | 22 | M-30 | 7308 | 23 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 24095 | 1003 | 2017-02-06 07:45:00 | 6 | M-30 | 7296 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 24309 | 1003 | 2017-02-14 07:45:00 | 14 | M-30 | 7296 | 24 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25687 | 1003 | 2017-02-21 08:30:00 | 21 | M-30 | 7272 | 24 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 44766 | 1018 | 2017-02-28 15:15:00 | 28 | M-30 | 7272 | 16 | 5 | (AFOROS) Pa Castellana N-S - Eduardo Dato-Marq… | -3.724722 | 40.421812 |
| 23732 | 1003 | 2017-02-01 07:30:00 | 1 | M-30 | 7272 | 23 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23728 | 1003 | 2017-02-01 07:15:00 | 1 | M-30 | 7260 | 23 | 4 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 24162 | 1003 | 2017-02-06 07:30:00 | 6 | M-30 | 7248 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23936 | 1003 | 2017-02-02 07:45:00 | 2 | M-30 | 7224 | 26 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 25675 | 1003 | 2017-02-21 08:00:00 | 21 | M-30 | 7224 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| … | … | … | … | … | … | … | … | … | … | … |
| 45994 | 1016 | 2017-02-25 10:15:00 | 25 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 436 | 1010 | 2017-02-06 02:30:00 | 6 | M-30 | 0 | 0 | 5 | Pl. Santa Barbara S-N - Serrano Anguita-Pl. A… | -3.725040 | 40.416313 |
| 47647 | 1016 | 2017-02-28 03:00:00 | 28 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 47649 | 1016 | 2017-02-28 03:15:00 | 28 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 47652 | 1016 | 2017-02-28 03:45:00 | 28 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 47653 | 1016 | 2017-02-28 04:00:00 | 28 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 47655 | 1016 | 2017-02-28 04:30:00 | 28 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 47657 | 1016 | 2017-02-28 05:00:00 | 28 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 6938 | 7014 | 2017-02-17 00:00:00 | 17 | M-30 | 0 | 0 | 1 | Jose Ortega y Gasset E-O - Montesa-Martires Co… | -3.672344 | 40.484965 |
| 6940 | 7014 | 2017-02-17 00:15:00 | 17 | M-30 | 0 | 0 | 3 | Jose Ortega y Gasset E-O - Montesa-Martires Co… | -3.672344 | 40.484965 |
| 47658 | 1016 | 2017-02-28 05:15:00 | 28 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 17833 | 6716 | 2017-02-23 02:30:00 | 23 | M-30 | 0 | 0 | 15 | (AFOROS) Velazquez, 24 S-N - Jorge Juan-Goya | -3.671378 | 40.471123 |
| 6951 | 7014 | 2017-02-16 06:15:00 | 16 | M-30 | 0 | 0 | 4 | Jose Ortega y Gasset E-O - Montesa-Martires Co… | -3.672344 | 40.484965 |
| 11843 | 7012 | 2017-02-19 03:15:00 | 19 | M-30 | 0 | 0 | 3 | Jose Ortega y Gasset O-E - Gral. Diaz Polier-C… | -3.763542 | 40.399098 |
| 466 | 1010 | 2017-02-06 03:30:00 | 6 | M-30 | 0 | 0 | 5 | Pl. Santa Barbara S-N - Serrano Anguita-Pl. A… | -3.725040 | 40.416313 |
| 6632 | 7014 | 2017-02-14 02:15:00 | 14 | M-30 | 0 | 0 | 4 | Jose Ortega y Gasset E-O - Montesa-Martires Co… | -3.672344 | 40.484965 |
| 11837 | 7012 | 2017-02-19 03:00:00 | 19 | M-30 | 0 | 0 | 1 | Jose Ortega y Gasset O-E - Gral. Diaz Polier-C… | -3.763542 | 40.399098 |
| 47792 | 1015 | 2017-02-01 03:30:00 | 1 | M-30 | 0 | 0 | 5 | P1 Castellana S-N - Pl. Colon-Hermosilla | -3.721861 | 40.420521 |
| 57572 | 1011 | 2017-02-20 05:00:00 | 20 | M-30 | 0 | 0 | 5 | (AFOROS) Genova O-E - General Castagos-Pl. Colon | -3.723105 | 40.418221 |
| 19548 | 7013 | 2017-02-14 02:15:00 | 14 | M-30 | 0 | 0 | 4 | Jose Ortega y Gasset, 67 E-O - Alcantara-Cond… | -3.672427 | 40.484974 |
| 55445 | 1012 | 2017-02-24 02:30:00 | 24 | M-30 | 0 | 0 | 5 | P0 Recoletos N-S - (PL. Colon-BErbara Braga… | -3.722132 | 40.419859 |
| 45373 | 1016 | 2017-02-06 01:45:00 | 6 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |
| 19554 | 7013 | 2017-02-19 03:15:00 | 19 | M-30 | 0 | 0 | 3 | Jose Ortega y Gasset, 67 E-O - Alcantara-Cond… | -3.672427 | 40.484974 |
| 17719 | 6716 | 2017-02-22 04:00:00 | 22 | M-30 | 0 | 0 | 1 | (AFOROS) Velazquez, 24 S-N - Jorge Juan-Goya | -3.671378 | 40.471123 |
| 36235 | 1001 | 2017-02-21 03:15:00 | 21 | M-30 | 0 | 0 | 5 | Jose Ortega y Gasset E-O - PA Castellana-Serrano | -3.740747 | 40.409738 |
| 36231 | 1001 | 2017-02-21 03:00:00 | 21 | M-30 | 0 | 0 | 5 | Jose Ortega y Gasset E-O - PA Castellana-Serrano | -3.740747 | 40.409738 |
| 36220 | 1001 | 2017-02-21 02:45:00 | 21 | M-30 | 0 | 0 | 5 | Jose Ortega y Gasset E-O - PA Castellana-Serrano | -3.740747 | 40.409738 |
| 55388 | 1012 | 2017-02-23 05:00:00 | 23 | M-30 | 0 | 0 | 5 | P0 Recoletos N-S - (PL. Colon-BErbara Braga… | -3.722132 | 40.419859 |
| 47986 | 1015 | 2017-02-07 04:00:00 | 7 | M-30 | 0 | 0 | 5 | P1 Castellana S-N - Pl. Colon-Hermosilla | -3.721861 | 40.420521 |
| 45546 | 1016 | 2017-02-07 05:30:00 | 7 | M-30 | 0 | 0 | 5 | Marques de la Ensenada N-S - Orellana - Barbar… | -3.721951 | 40.420470 |

58429 rows × 10 columns

## Grupos y operaciones de agrupación[¶](#Grupos-y-operaciones-de-agrupación)

* Supongamos que queremos calcular la intensidad máxima registrada por tipo de elemento (M-30 o Urbano)
* Supongamos que queremos calcular la intensidad media de tráfico por estación.
* Supongamos que queremos calcular la intensidad media de tráfico por día del mes.

Pandas proporciona el método groupby para construir grupos sobre los cuales se aplica posteriormente alguna operación de agregación, como la media, la suma, máximo, etc.

In [46]:

# intensidad máxima por tipo de elemento  
tf.groupby(['tipo\_elem']).intensidad.max().reset\_index()

Out[46]:

|  |  |  |
| --- | --- | --- |
|  | tipo\_elem | intensidad |
| 0 | M-30 | 9232 |
| 1 | Urbanos | 99999 |

In [47]:

# intensidad media por tipo de elemento  
tf.groupby(['tipo\_elem']).intensidad.mean().reset\_index()

Out[47]:

|  |  |  |
| --- | --- | --- |
|  | tipo\_elem | intensidad |
| 0 | M-30 | 1448.641587 |
| 1 | Urbanos | 368.434780 |

In [48]:

# los dos valores, el máximo y la media  
tf.groupby(['tipo\_elem']).intensidad.agg([max, np.mean])

Out[48]:

max

mean

tipo\_elem

M-30

9232

1448.641587

Urbanos

99999

368.434780

Intensidad media por estación. Además ordenamos los valores obtenidos de forma descendente.

In [49]:

tf.groupby(['idelem']).intensidad.mean().sort\_values(ascending = False)

Out[49]:

idelem  
3819 4456.097987  
6679 4083.683408  
6666 3931.986235  
6747 3852.108340  
6733 3773.160424  
6725 3744.718524  
6676 3677.470961  
3820 3672.540984  
6846 3667.480655  
6756 3597.010789  
6727 3519.084720  
6642 3485.329081  
3490 3393.583333  
6842 3387.531994  
6737 3377.491071  
6845 3376.241443  
3823 3316.648801  
6715 3315.266017  
3799 3301.719494  
6843 3290.052455  
6685 3247.393376  
6662 3201.364083  
3815 3175.915551  
6789 3103.891686  
6665 3099.061384  
6663 3086.258752  
3491 3083.810640  
6847 3067.225818  
6683 3060.428359  
6654 3057.895089  
 ...   
6895 17.471354  
6187 16.473885  
5029 16.318182  
5187 16.298578  
5993 16.076299  
7000 15.666667  
6537 15.555556  
5033 15.530000  
6236 15.300000  
5089 15.226316  
6539 14.750220  
3928 13.715711  
10135 13.710000  
6626 13.514539  
4914 13.144873  
5032 11.596875  
5173 10.637191  
6268 8.888889  
6547 8.778409  
3680 4.833333  
3861 0.000000  
7060 0.000000  
4160 0.000000  
6396 0.000000  
6267 0.000000  
6298 0.000000  
4202 0.000000  
3526 0.000000  
6822 0.000000  
4384 0.000000  
Name: intensidad, dtype: float64

**Agrupar por varios campos**

Para calcular la intensidad media de tráfico por día del mes, tenemos que agrupar por dos columnas:

In [50]:

tf\_m30.groupby(['idelem', 'dia']).intensidad.mean().reset\_index()

Out[50]:

|  |  |  |  |
| --- | --- | --- | --- |
|  | idelem | dia | intensidad |
| 0 | 1001 | 1 | 1854.416667 |
| 1 | 1001 | 2 | 1903.354167 |
| 2 | 1001 | 3 | 2023.281250 |
| 3 | 1001 | 4 | 1656.604167 |
| 4 | 1001 | 5 | 1049.250000 |
| 5 | 1001 | 6 | 1887.500000 |
| 6 | 1001 | 7 | 1705.750000 |
| 7 | 1001 | 8 | 1954.375000 |
| 8 | 1001 | 9 | 2241.500000 |
| 9 | 1001 | 10 | 2064.375000 |
| 10 | 1001 | 11 | 1728.000000 |
| 11 | 1001 | 12 | 1417.770833 |
| 12 | 1001 | 13 | 2167.541667 |
| 13 | 1001 | 14 | 1954.093750 |
| 14 | 1001 | 15 | 2112.375000 |
| 15 | 1001 | 16 | 2243.281250 |
| 16 | 1001 | 17 | 2266.125000 |
| 17 | 1001 | 18 | 1730.312500 |
| 18 | 1001 | 19 | 1438.916667 |
| 19 | 1001 | 20 | 1939.708333 |
| 20 | 1001 | 21 | 2049.020833 |
| 21 | 1001 | 22 | 2197.125000 |
| 22 | 1001 | 23 | 2038.750000 |
| 23 | 1001 | 24 | 2165.406250 |
| 24 | 1001 | 25 | 584.656250 |
| 25 | 1001 | 26 | 1481.375000 |
| 26 | 1001 | 27 | 1773.854167 |
| 27 | 1001 | 28 | 1886.250000 |
| 28 | 1002 | 1 | 2249.500000 |
| 29 | 1002 | 2 | 2258.250000 |
| … | … | … | … |
| 10577 | 10177 | 27 | 1404.041667 |
| 10578 | 10177 | 28 | 1394.052083 |
| 10579 | 10180 | 1 | 1787.729167 |
| 10580 | 10180 | 2 | 1888.760417 |
| 10581 | 10180 | 3 | 1828.531250 |
| 10582 | 10180 | 4 | 1170.031250 |
| 10583 | 10180 | 5 | 1088.197917 |
| 10584 | 10180 | 6 | 1750.687500 |
| 10585 | 10180 | 7 | 1787.270833 |
| 10586 | 10180 | 8 | 1837.989583 |
| 10587 | 10180 | 9 | 1792.937500 |
| 10588 | 10180 | 10 | 1793.572917 |
| 10589 | 10180 | 11 | 1112.562500 |
| 10590 | 10180 | 12 | 1059.406250 |
| 10591 | 10180 | 13 | 1695.781250 |
| 10592 | 10180 | 14 | 1798.937500 |
| 10593 | 10180 | 15 | 1843.177083 |
| 10594 | 10180 | 16 | 1882.270833 |
| 10595 | 10180 | 17 | 1853.843750 |
| 10596 | 10180 | 18 | 1272.114583 |
| 10597 | 10180 | 19 | 1220.750000 |
| 10598 | 10180 | 20 | 1767.072917 |
| 10599 | 10180 | 21 | 1820.541667 |
| 10600 | 10180 | 22 | 1863.010417 |
| 10601 | 10180 | 23 | 1864.489583 |
| 10602 | 10180 | 24 | 1949.531250 |
| 10603 | 10180 | 25 | 1473.468750 |
| 10604 | 10180 | 26 | 1291.708333 |
| 10605 | 10180 | 27 | 1789.656250 |
| 10606 | 10180 | 28 | 1843.645833 |

10607 rows × 3 columns

Cuando se trabaja con fechas, podemos agrupar los datos por minutos, horas, días, años, etc. En este caso, vamos a agrupar los datos por horas y calculamos la media de itensidad de todas las estaciones por hora. Para ellos usamos el método resermple de Pandas.

**Agrupar datos teniendo en cuenta la fecha**

In [51]:

tf.resample('H', on='fecha').apply(np.mean).intensidad.reset\_index().head()

Out[51]:

|  |  |  |
| --- | --- | --- |
|  | fecha | intensidad |
| 0 | 2017-02-01 00:00:00 | 164.886881 |
| 1 | 2017-02-01 01:00:00 | 84.339541 |
| 2 | 2017-02-01 02:00:00 | 51.511014 |
| 3 | 2017-02-01 03:00:00 | 39.851909 |
| 4 | 2017-02-01 04:00:00 | 39.069174 |

**Agrupar por varios campos**

In [52]:

tf.groupby(['fecha', 'idelem']).intensidad.mean().reset\_index()

Out[52]:

|  |  |  |  |
| --- | --- | --- | --- |
|  | fecha | idelem | intensidad |
| 0 | 2017-02-01 00:00:00 | 1001 | 372 |
| 1 | 2017-02-01 00:00:00 | 1002 | 684 |
| 2 | 2017-02-01 00:00:00 | 1003 | 600 |
| 3 | 2017-02-01 00:00:00 | 1006 | 744 |
| 4 | 2017-02-01 00:00:00 | 1009 | 648 |
| 5 | 2017-02-01 00:00:00 | 1010 | 1116 |
| 6 | 2017-02-01 00:00:00 | 1011 | 672 |
| 7 | 2017-02-01 00:00:00 | 1012 | 384 |
| 8 | 2017-02-01 00:00:00 | 1013 | 780 |
| 9 | 2017-02-01 00:00:00 | 1014 | 420 |
| 10 | 2017-02-01 00:00:00 | 1015 | 600 |
| 11 | 2017-02-01 00:00:00 | 1016 | 192 |
| 12 | 2017-02-01 00:00:00 | 1017 | 768 |
| 13 | 2017-02-01 00:00:00 | 1018 | 1068 |
| 14 | 2017-02-01 00:00:00 | 1019 | 1128 |
| 15 | 2017-02-01 00:00:00 | 1020 | 792 |
| 16 | 2017-02-01 00:00:00 | 1021 | 828 |
| 17 | 2017-02-01 00:00:00 | 1022 | 192 |
| 18 | 2017-02-01 00:00:00 | 1023 | 300 |
| 19 | 2017-02-01 00:00:00 | 1024 | 60 |
| 20 | 2017-02-01 00:00:00 | 1025 | 36 |
| 21 | 2017-02-01 00:00:00 | 1026 | 72 |
| 22 | 2017-02-01 00:00:00 | 1027 | 84 |
| 23 | 2017-02-01 00:00:00 | 1028 | 444 |
| 24 | 2017-02-01 00:00:00 | 1029 | 108 |
| 25 | 2017-02-01 00:00:00 | 1030 | 240 |
| 26 | 2017-02-01 00:00:00 | 1031 | 348 |
| 27 | 2017-02-01 00:00:00 | 1032 | 84 |
| 28 | 2017-02-01 00:00:00 | 1033 | 240 |
| 29 | 2017-02-01 00:00:00 | 1034 | 192 |
| … | … | … | … |
| 8823487 | 2017-02-28 23:45:00 | 10143 | 25 |
| 8823488 | 2017-02-28 23:45:00 | 10144 | 35 |
| 8823489 | 2017-02-28 23:45:00 | 10145 | 26 |
| 8823490 | 2017-02-28 23:45:00 | 10146 | 198 |
| 8823491 | 2017-02-28 23:45:00 | 10147 | 179 |
| 8823492 | 2017-02-28 23:45:00 | 10148 | 209 |
| 8823493 | 2017-02-28 23:45:00 | 10149 | 172 |
| 8823494 | 2017-02-28 23:45:00 | 10150 | 54 |
| 8823495 | 2017-02-28 23:45:00 | 10151 | 170 |
| 8823496 | 2017-02-28 23:45:00 | 10152 | 130 |
| 8823497 | 2017-02-28 23:45:00 | 10153 | 49 |
| 8823498 | 2017-02-28 23:45:00 | 10154 | 80 |
| 8823499 | 2017-02-28 23:45:00 | 10155 | 47 |
| 8823500 | 2017-02-28 23:45:00 | 10156 | 41 |
| 8823501 | 2017-02-28 23:45:00 | 10157 | 214 |
| 8823502 | 2017-02-28 23:45:00 | 10158 | 172 |
| 8823503 | 2017-02-28 23:45:00 | 10159 | 325 |
| 8823504 | 2017-02-28 23:45:00 | 10160 | 227 |
| 8823505 | 2017-02-28 23:45:00 | 10161 | 471 |
| 8823506 | 2017-02-28 23:45:00 | 10162 | 123 |
| 8823507 | 2017-02-28 23:45:00 | 10163 | 248 |
| 8823508 | 2017-02-28 23:45:00 | 10164 | 150 |
| 8823509 | 2017-02-28 23:45:00 | 10165 | 33 |
| 8823510 | 2017-02-28 23:45:00 | 10166 | 72 |
| 8823511 | 2017-02-28 23:45:00 | 10167 | 65 |
| 8823512 | 2017-02-28 23:45:00 | 10168 | 214 |
| 8823513 | 2017-02-28 23:45:00 | 10169 | 89 |
| 8823514 | 2017-02-28 23:45:00 | 10170 | 68 |
| 8823515 | 2017-02-28 23:45:00 | 10177 | 488 |
| 8823516 | 2017-02-28 23:45:00 | 10180 | 548 |

8823517 rows × 3 columns

## Exportar datos a formatos excel, csv, txt, json[¶](#Exportar-datos-a-formatos-excel,-csv,-txt,-json)

In [53]:

tf\_m30.to\_excel('../datos/trafico\_m30.xlsx')

In [54]:

tf\_m30.to\_csv('../datos/trafico\_m30.csv')

## Visualización[¶](#Visualización)

Existen varias librerías para realizar las tareas de visualización:

### Matplotlib[¶](#Matplotlib)

Matplotlib es una de las librerías en Python más utilizadas en ciencia de datos.

* Una de las caraterísticas que hace que tenga éxito es **la facilidad de uso**
* Visualizaciones de datos con **muy pocas líneas de código**
* Los gráficos se puedan incluir en cualquier proyecto web
* Diagramas de barras

Gráficos de áreas, histogramas, visualizaciones de líneas, barras, diagramas de dispersión… matplotlib es una de las librerías en Python más utilizadas en ciencia de datos.

[Librería gráfica matplotlib](http://matplotlib.org/) (<http://matplotlib.org/>)

In [55]:

import matplotlib.pyplot as plt  
%matplotlib inline

#### Diagrama de barras[¶](#Diagrama-de-barras)

Representación de la intensidad media por día del mes.

In [56]:

# intensidad media de tráfico recogida por cada estación cada día de la semana  
# preparo los datos: agrupo por día y calculo la media de la intensidad  
pordias = tf\_m30.groupby(['dia']).intensidad.mean().reset\_index()

In [57]:

fig, ax = plt.subplots(nrows = 1, figsize=(10,5))  
ax.bar(pordias.dia, pordias.intensidad)   
ax.legend(loc=0)  
  
ax.set\_title('Diagrama de barras con Matplotlib')  
ax.set\_xlabel('Dias del mes')  
ax.set\_ylabel('Intensidad media');

C:\Users\Yoli\Anaconda3\lib\site-packages\matplotlib\axes\\_axes.py:545: UserWarning: No labelled objects found. Use label='...' kwarg on individual plots.  
 warnings.warn("No labelled objects found. "

![](data:image/png;base64;base64,)

### Seaborn[¶](#Seaborn)

Seaborn es una librería de visualización de datos en Python basada en matplotlib. Sus características:

* Mejora el diseño de matplotlib.
* Dispone de herramientas para la elección de paletas de colores.
* Funciones para comparar subconjuntos de datos.
* Herramientas para adaptar y visualizar modelos de regresión lineal.
* Funciones para visualizar matrices de datos.
* Uso de algoritmos de clustering.
* Posibilidad de establecer series temporales estadísticas con los datos.

[Seaborn](https://seaborn.pydata.org/) (<https://seaborn.pydata.org/>)

In [58]:

# módulos necesarios  
import seaborn as sns  
import numpy as np  
import matplotlib.pyplot as plt  
%matplotlib inline

#### Diagrama de barras con Seaborn[¶](#Diagrama-de-barras-con-Seaborn)

In [59]:

sns.set(style="white", context="talk")  
  
# figuras  
f, ax1 = plt.subplots(1, figsize=(8, 6))  
  
# Ejemplo 1  
sns.barplot(pordias.dia, pordias.intensidad, palette="BuGn\_d", ax=ax1)  
ax1.set\_ylabel("Intensidad media");

![](data:image/png;base64;base64,)

#### Diagramas de calor[¶](#Diagramas-de-calor)

Creamos un dataframe aplicando el método pivot\_table. La nueva tabla se crea a partir de 3 columnas.

In [60]:

tf\_p = tf\_m30.pivot\_table(index = "idelem", columns = "dia", values = "intensidad")

In [61]:

tf\_p.head(10)

Out[61]:

dia

1

2

3

4

5

6

7

8

9

10

…

19

20

21

22

23

24

25

26

27

28

idelem

1001

1854.416667

1903.354167

2023.28125

1656.604167

1049.250000

1887.500

1705.750000

1954.375000

2241.500000

2064.3750

…

1438.916667

1939.708333

2049.020833

2197.12500

2038.75000

2165.406250

584.65625

1481.375000

1773.854167

1886.250

1002

2249.500000

2258.250000

2328.12500

1716.229167

1492.500000

2238.000

2180.312500

2169.375000

2313.000000

2353.5000

…

1551.666667

2150.833333

2229.354167

2287.18750

2312.84375

2405.364583

527.65625

1600.041667

2271.562500

2302.000

1003

2968.041667

3023.416667

3073.25000

2276.666667

1969.125000

2935.375

2881.062500

2849.875000

3024.125000

3075.3750

…

2050.666667

2872.208333

2953.104167

3018.62500

3093.25000

3183.229167

821.96875

2260.645833

2981.208333

3013.375

1006

2313.437500

2329.791667

2327.15625

1753.468750

1530.750000

2256.375

2192.812500

2271.000000

2359.250000

2497.6250

…

1574.875000

2206.875000

2247.645833

2287.65625

2376.53125

2484.968750

677.84375

1700.062500

2315.895833

2339.125

1009

2329.875000

2361.708333

2377.68750

1753.625000

1564.500000

2265.875

2303.625000

2269.500000

2389.250000

2388.6250

…

1566.208333

2258.666667

2309.770833

2376.03125

2382.90625

2442.187500

852.96875

1721.104167

2345.427083

2386.875

1010

2265.062500

2319.687500

2308.71875

1694.500000

1419.083333

2466.750

2030.458333

2568.166667

2929.833333

2927.4375

…

1998.166667

2000.239583

1748.666667

1164.37500

1696.50000

1820.291667

479.50000

743.166667

1685.270833

1747.000

1011

1414.312500

1424.062500

1451.84375

1006.260417

931.625000

1400.750

1374.187500

1395.875000

1468.625000

1433.1250

…

965.875000

1420.000000

1425.729167

1423.87500

1411.40625

1498.270833

447.71875

1052.208333

1420.302083

1485.375

1012

647.354167

664.562500

717.84375

509.864583

496.125000

670.375

661.125000

689.125000

708.125000

767.3750

…

500.541667

679.541667

667.270833

673.96875

700.12500

763.114583

274.28125

556.250000

696.000000

687.875

1013

1983.791667

1979.041667

1996.90625

1359.802083

1266.250000

1912.500

1994.437500

1985.625000

2027.875000

1962.6250

…

1401.375000

1984.375000

2038.916667

2052.81250

2051.62500

2049.479167

809.34375

1442.270833

1997.864583

2006.500

1014

1165.604167

1143.791667

1154.96875

837.614583

731.250000

1111.500

1107.187500

1130.500000

1169.500000

1149.5000

…

785.833333

1133.250000

1152.958333

1145.87500

1166.12500

1213.302083

316.59375

896.666667

1162.145833

1175.375

10 rows × 28 columns

Construimos un daigrama de calor de las 10 primeras estaciones:

In [62]:

fig, g = plt.subplots(figsize=(15, 5))  
# mapa de calor  
sns.heatmap(tf\_p.head(10), annot=False, vmax=3000, linewidths=1, ax = g);  
  
# decoración  
g.set\_title('Intensidad de tráfico - Febrero 2017', fontsize = 23);  
g.axes.xaxis.set\_label\_text("Días de la semana", fontdict={"size":16})  
g.axes.yaxis.set\_label\_text("Estaciones de tráfico", fontdict={"size":16})  
plt.yticks( rotation='horizontal', fontsize = 14);

![](data:image/png;base64;base64,)

¿Cuál es el nombre de las estación 1012?

In [63]:

merged[merged.idelem == 1003]

Out[63]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | idelem | fecha | dia | tipo\_elem | intensidad | ocupacion | periodo\_integracion | nombre | Xcoord | Ycoord |
| 23485 | 1003 | 2017-02-22 15:15:00 | 22 | M-30 | 3612 | 10 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23486 | 1003 | 2017-02-01 21:30:00 | 1 | M-30 | 2988 | 9 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23487 | 1003 | 2017-02-01 09:00:00 | 1 | M-30 | 6564 | 27 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23488 | 1003 | 2017-02-14 23:15:00 | 14 | M-30 | 1368 | 4 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23489 | 1003 | 2017-02-14 05:45:00 | 14 | M-30 | 348 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23490 | 1003 | 2017-02-15 17:30:00 | 15 | M-30 | 3720 | 13 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23491 | 1003 | 2017-02-25 09:45:00 | 25 | M-30 | 360 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23492 | 1003 | 2017-02-02 10:15:00 | 2 | M-30 | 5124 | 17 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23493 | 1003 | 2017-02-04 14:30:00 | 4 | M-30 | 4836 | 13 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23494 | 1003 | 2017-02-05 19:45:00 | 5 | M-30 | 2868 | 9 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23495 | 1003 | 2017-02-01 09:15:00 | 1 | M-30 | 6108 | 21 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23496 | 1003 | 2017-02-05 04:30:00 | 5 | M-30 | 540 | 2 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23497 | 1003 | 2017-02-04 14:45:00 | 4 | M-30 | 4500 | 12 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23498 | 1003 | 2017-02-02 22:45:00 | 2 | M-30 | 2124 | 7 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23499 | 1003 | 2017-02-01 22:00:00 | 1 | M-30 | 2208 | 7 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23500 | 1003 | 2017-02-01 09:30:00 | 1 | M-30 | 6432 | 22 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23501 | 1003 | 2017-02-01 09:45:00 | 1 | M-30 | 6072 | 19 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23502 | 1003 | 2017-02-01 21:45:00 | 1 | M-30 | 2640 | 10 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23503 | 1003 | 2017-02-04 01:15:00 | 4 | M-30 | 612 | 2 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23504 | 1003 | 2017-02-03 12:30:00 | 3 | M-30 | 3276 | 10 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23505 | 1003 | 2017-02-03 12:15:00 | 3 | M-30 | 3816 | 10 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23506 | 1003 | 2017-02-02 10:30:00 | 2 | M-30 | 5364 | 17 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23507 | 1003 | 2017-02-01 22:15:00 | 1 | M-30 | 1704 | 5 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23508 | 1003 | 2017-02-04 15:00:00 | 4 | M-30 | 4000 | 11 | 3 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23509 | 1003 | 2017-02-02 23:00:00 | 2 | M-30 | 1608 | 5 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23510 | 1003 | 2017-02-01 10:15:00 | 1 | M-30 | 4620 | 15 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23511 | 1003 | 2017-02-06 11:45:00 | 6 | M-30 | 3780 | 12 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23512 | 1003 | 2017-02-05 04:45:00 | 5 | M-30 | 432 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23513 | 1003 | 2017-02-01 10:00:00 | 1 | M-30 | 5748 | 18 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 23514 | 1003 | 2017-02-05 05:00:00 | 5 | M-30 | 456 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| … | … | … | … | … | … | … | … | … | … | … |
| 26143 | 1003 | 2017-02-28 23:15:00 | 28 | M-30 | 1332 | 4 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26144 | 1003 | 2017-02-28 03:00:00 | 28 | M-30 | 168 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26145 | 1003 | 2017-02-28 23:30:00 | 28 | M-30 | 1044 | 4 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26146 | 1003 | 2017-02-28 23:45:00 | 28 | M-30 | 876 | 2 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26147 | 1003 | 2017-02-28 03:15:00 | 28 | M-30 | 156 | 0 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26148 | 1003 | 2017-02-28 03:30:00 | 28 | M-30 | 168 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26149 | 1003 | 2017-02-28 03:45:00 | 28 | M-30 | 60 | -1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26150 | 1003 | 2017-02-28 04:00:00 | 28 | M-30 | 108 | 0 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26151 | 1003 | 2017-02-28 04:15:00 | 28 | M-30 | 204 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26152 | 1003 | 2017-02-28 04:30:00 | 28 | M-30 | 132 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26153 | 1003 | 2017-02-28 04:45:00 | 28 | M-30 | 168 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26154 | 1003 | 2017-02-28 05:00:00 | 28 | M-30 | 252 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26155 | 1003 | 2017-02-28 05:15:00 | 28 | M-30 | 348 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26156 | 1003 | 2017-02-28 05:30:00 | 28 | M-30 | 480 | 2 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26157 | 1003 | 2017-02-28 05:45:00 | 28 | M-30 | 324 | 1 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26158 | 1003 | 2017-02-28 06:00:00 | 28 | M-30 | 924 | 3 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26159 | 1003 | 2017-02-28 06:15:00 | 28 | M-30 | 1320 | 4 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26160 | 1003 | 2017-02-28 06:30:00 | 28 | M-30 | 2076 | 5 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26161 | 1003 | 2017-02-28 06:45:00 | 28 | M-30 | 3624 | 10 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26162 | 1003 | 2017-02-28 07:00:00 | 28 | M-30 | 5652 | 17 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26163 | 1003 | 2017-02-28 07:15:00 | 28 | M-30 | 6216 | 20 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26164 | 1003 | 2017-02-28 07:30:00 | 28 | M-30 | 7092 | 24 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26165 | 1003 | 2017-02-28 07:45:00 | 28 | M-30 | 7536 | 25 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26166 | 1003 | 2017-02-28 08:00:00 | 28 | M-30 | 6888 | 23 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26167 | 1003 | 2017-02-28 08:15:00 | 28 | M-30 | 6996 | 23 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26168 | 1003 | 2017-02-28 08:30:00 | 28 | M-30 | 6372 | 19 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26169 | 1003 | 2017-02-28 08:45:00 | 28 | M-30 | 6708 | 21 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26170 | 1003 | 2017-02-28 09:00:00 | 28 | M-30 | 6156 | 19 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26171 | 1003 | 2017-02-28 09:15:00 | 28 | M-30 | 5928 | 18 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |
| 26172 | 1003 | 2017-02-28 09:30:00 | 28 | M-30 | 6024 | 18 | 5 | P1 Recoletos N-S - Almirante-Prim | -3.746831 | 40.406825 |

2688 rows × 10 columns

### Bokeh[¶](#Bokeh)

Se trata de una librería al estilo JavaScript D3.js.

* Es una librería **interactiva**.
* Cada gráfico tiene asociado su código HTML para incluirlo en una página web.
* Consultar la galería <http://bokeh.pydata.org/en/latest/docs/gallery.html>

#### Series de tiempo[¶](#Series-de-tiempo)

Representación de la evolución de la media de la intensidad a lo largo del tiempo:

In [64]:

porfecha = tf\_m30.groupby(['fecha']).intensidad.mean().reset\_index()  
porfecha.head()

Out[64]:

|  |  |  |
| --- | --- | --- |
|  | fecha | intensidad |
| 0 | 2017-02-01 00:00:00 | 496.671088 |
| 1 | 2017-02-01 00:15:00 | 451.864721 |
| 2 | 2017-02-01 00:30:00 | 397.740053 |
| 3 | 2017-02-01 00:45:00 | 329.132626 |
| 4 | 2017-02-01 01:00:00 | 267.466844 |

In [65]:

from bokeh.plotting import figure, output\_notebook, show, output\_file  
output\_notebook() # para visualizar el gráfico en el notebook  
  
  
output\_file("datetime.html")  
p = figure(title="Series de tiempo", plot\_width=600, plot\_height=300, x\_axis\_type="datetime")  
p.line(porfecha.fecha, porfecha.intensidad, color='navy', alpha=0.8)  
  
show(p)

Loading BokehJS …

## Otra librería de Visualización: python-higthcharts[¶](#Otra-librería-de-Visualización:-python-higthcharts)

HighChart es una librería escrita en Javascript que permite la creación de gráficas. La librería ofrece un método fácil e interactivo para insertar graficas en su sitio web o aplicación web.

In [66]:

from highcharts import Highchart # import highchart library  
H = Highchart(width=650, height=400)  
data = [{  
 'name': "Microsoft Internet Explorer",  
 'y': 56.33,  
 'drilldown': "Microsoft Internet Explorer"  
 }, {  
 'name': "Chrome",  
 'y': 24.030000000000005,  
 'drilldown': "Chrome"  
 }, {  
 'name': "Firefox",  
 'y': 10.38,  
 'drilldown': "Firefox"  
 }, {  
 'name': "Safari",  
 'y': 4.77,  
 'drilldown': "Safari"  
 }, {  
 'name': "Opera",  
 'y': 0.9100000000000001,  
 'drilldown': "Opera"  
 }, {  
 'name': "Proprietary or Undetectable",  
 'y': 0.2,  
 'drilldown': None  
 }]  
  
data\_1 = [  
 ["v11.0", 24.13],  
 ["v8.0", 17.2],  
 ["v9.0", 8.11],  
 ["v10.0", 5.33],  
 ["v6.0", 1.06],  
 ["v7.0", 0.5]  
]  
  
data\_2 = [  
 ["v40.0", 5],  
 ["v41.0", 4.32],  
 ["v42.0", 3.68],  
 ["v39.0", 2.96],  
 ["v36.0", 2.53],  
 ["v43.0", 1.45],  
 ["v31.0", 1.24],  
 ["v35.0", 0.85],  
 ["v38.0", 0.6],  
 ["v32.0", 0.55],  
 ["v37.0", 0.38],  
 ["v33.0", 0.19],  
 ["v34.0", 0.14],  
 ["v30.0", 0.14]   
]  
  
data\_3 = [  
 ["v35", 2.76],  
 ["v36", 2.32],  
 ["v37", 2.31],  
 ["v34", 1.27],  
 ["v38", 1.02],  
 ["v31", 0.33],  
 ["v33", 0.22],  
 ["v32", 0.15]  
]  
data\_4 = [  
 ["v8.0", 2.56],  
 ["v7.1", 0.77],  
 ["v5.1", 0.42],  
 ["v5.0", 0.3],  
 ["v6.1", 0.29],  
 ["v7.0", 0.26],  
 ["v6.2", 0.17]  
]  
  
data\_5 = [  
 ["v12.x", 0.34],  
 ["v28", 0.24],  
 ["v27", 0.17],  
 ["v29", 0.16]  
]  
  
# Añadimos los datos  
H.add\_data\_set(data, series\_type = 'column', name = "Marcas", colorByPoint= True)  
  
H.add\_drilldown\_data\_set(data\_1, 'column', 'Microsoft Internet Explorer', name='Microsoft Internet Explorer' )  
H.add\_drilldown\_data\_set(data\_2, 'column', 'Chrome', name='Chrome')  
H.add\_drilldown\_data\_set(data\_3, 'column', 'Firefox', name='Firefox')  
H.add\_drilldown\_data\_set(data\_4, 'column', 'Safari', name='Safari')  
H.add\_drilldown\_data\_set(data\_5, 'column', 'Opera', name='Opera')  
  
# Especificamos las opciones del grágico  
  
  
options = {  
 'chart': {  
 'type': 'column'  
 },  
 'title': {  
 'text': 'Browser market shares. January, 2015 to May, 2015'  
 },  
 'subtitle': {  
 'text': 'Click the columns to view versions. Source: <a href="http://netmarketshare.com">netmarketshare.com</a>.'  
 },  
 'xAxis': {  
 'type': 'category'  
 },  
 'yAxis': {  
 'title': {  
 'text': 'Total percent market share'  
 }  
  
 },  
 'legend': {  
 'enabled': False  
 },  
 'plotOptions': {  
 'series': {  
 'borderWidth': 0,  
 'dataLabels': {  
 'enabled': True,  
 'format': '{point.y:.1f}%'  
 }  
 }  
 },  
  
 'tooltip': {  
 'headerFormat': '<span style="font-size:11px">{series.name}</span><br>',  
 'pointFormat': '<span style="color:{point.color}">{point.name}</span>: <b>{point.y:.2f}%</b> of total<br/>'  
 },   
   
}   
H.set\_dict\_options(options)  
H

Out[66]:

## References[¶](#References)

* [Tutorial de Python. Por Guido Van Rossum](http://docs.python.org.ar/tutorial/pdfs/TutorialPython3.pdf)
* [Python for Data Analysis](http://shop.oreilly.com/product/0636920023784.do) (<http://shop.oreilly.com/product/0636920023784.do>)
* [Librería gráfica matplotlib](http://matplotlib.org/) (<http://matplotlib.org/>)
* [Seaborn](https://seaborn.pydata.org/) (<https://seaborn.pydata.org/>)
* [Bokeh](http://bokeh.pydata.org/en/latest/) (<http://bokeh.pydata.org/en/latest/>)