

# Python para Análisis de Datos

Módulo 05

# Pandas

# Fechas

Para representar **fechas y horas** se utiliza un tipo de dato especial: `datetime64`. Y para representar **intervalos de tiempo** se usa `timedelta64`. Son tipos de datos que incorporan características y operaciones propias de trabajar y medir tiempos.

Usualmente cuando leemos un archivo las fechas se representan como strings. Para convertirlo a `datetime64` se usa la función `pd.to_datetime`, que permite controlar el formato con el que se va a leer la información, que puede venir de diferentes formas: `yyyy-mm-dd`, `dd/mm/yy`, `Month dd, yyyy`, etc.



## Fechas

Para esta sección vamos a usar de ejemplo el archivo **GoldPrice.csv**. Este archivo contiene casi 10 años de registros en el precio del oro. La primera columna tiene strings que representan las fechas en el formato "Sep 11, 2020". Podemos convertir la columna al tipo de dato datetime con `pd.to_datetime`.

```
gold = pd.read_csv("GoldPrice.csv")  
gold.dtypes
```

```
Date      object  
Price     float64  
Open      float64  
High      float64  
Low       float64  
Chg%      float64  
dtype: object
```

```
gold["Date"] = pd.to_datetime(gold["Date"])  
gold.dtypes
```

```
Date      datetime64[ns]  
Price     float64  
Open      float64  
High      float64  
Low       float64  
Chg%      float64  
dtype: object
```

## Fechas

Los objetos de tipo datetime tienen varios atributos, como year, month, day, hour, etc. Para acceder a los atributos y métodos propios del tipo de dato datetime se utiliza el atributo dt.

```
gold["Date"].dt.year
```

```
0      2020
1      2020
2      2020
3      2020
4      2020
...
2526   2011
2527   2011
2528   2011
```

```
gold["Date"].dt.month
```

```
0      9
1      9
2      9
3      9
4      9
...
2526    1
2527    1
2528    1
```

```
gold["Date"].dt.day
```

```
0      11
1      10
2       9
3       8
4       7
...
2526    7
2527    6
2528    5
```

## Fechas

Podemos encontrar la fecha más antigua con `min` y la más reciente con `max`. La resta de fechas da una instancia de `Timedelta`, que representa intervalos de tiempo. No se puede sumar dos fechas, pero sí se puede sumar un `Datetime` con un `Timedelta`.

```
gold["Date"].min(), gold["Date"].max()  
  
(Timestamp('2011-01-03 00:00:00'), Timestamp('2020-09-11 00:00:00'))
```

```
gold["Date"].max() - gold["Date"].min()  
  
Timedelta('3539 days 00:00:00')
```

## Fechas

Cuando tenemos datos relacionados a fechas, podemos utilizar la serie de tiempo como índice. Esto le da mucha funcionalidad ya que podemos seleccionar datos refiriéndonos a las fechas.

Para utilizar la columna de fechas como índice podemos usar `set_index`. En este caso pandas crea un tipo de dato especial, el `DatetimeIndex`. Como sólo se trata de fechas podemos acceder a los métodos directamente, sin pasar por el atributo `dt`.

Ahora podemos seleccionar los datos según criterios temporales, como todos los datos de un año o de todos los lunes, usando indexing y filtros.

## Fechas

```
gold
```

	Date	Price	Open	High	Low	Chg%
0	2020-09-11	1957.35	1952.55	1963.3	1944.35	-0.0035
1	2020-09-10	1964.30	1955.30	1975.2	1948.60	0.0048

```
gold.set_index("Date", inplace=True)  
gold
```

	Price	Open	High	Low	Chg%
Date					
2020-09-11	1957.35	1952.55	1963.3	1944.35	-0.0035
2020-09-10	1964.30	1955.30	1975.2	1948.60	0.0048

```
gold.index
```

```
DatetimeIndex(['2020-09-11', '2020-09-10', '2020-09-09', '2020-09-08',  
               '2020-09-07', '2020-09-06', '2020-09-04', '2020-09-03',
```



## Fechas

```
# seleccionar el año 2016  
gold["2016"]
```

	Price	Open	High	Low	Chg%
Date					
2016-12-30	1234.50	1239.0	1239.00	1239.0	-0.0054
2016-12-29	1241.20	1241.2	1241.20	1241.2	0.0141

```
# registros posteriores a una fecha  
gold[gold.index > "2018-4-20"]
```

	Price	Open	High	Low	Chg%
Date					
2020-09-11	1957.35	1952.55	1963.3	1944.35	-0.0035
2020-09-10	1964.30	1955.30	1975.2	1948.60	0.0048

## Fechas

```
# seleccionar todos los lunes del dataset  
gold[gold.index.dayofweek == 0]
```

	Price	Open	High	Low	Chg%
Date					
2020-09-07	1937.1	1940.7	1947.4	1930.45	-0.0018
2020-08-31	1978.6	1973.9	1985.8	1962.30	0.0019
2020-08-24	1939.2	1947.9	1970.3	1930.80	-0.0040

# ¡Muchas gracias!

¡Sigamos trabajando!