

Funciones básicas

```
df = pd.read_csv("df.csv", parse_dates = ["col_name"])
```

Cargar csv y convertir una columna a formato de fecha

```
df.to_period("M")
```

Convert datetime index of dataframe to YYYY-MM

```
pd.Timestamp("1/2/2019")
```

```
pd.Timestamp(2019, 2, 1)
```

```
pd.Timestamp(date(2019, 8, 26))
```

```
pd.Timestamp(2019, 2, 1, 16, 17, 22)
```

```
pd.Timestamp(year = 2019, month = 10, day = 28, hour = 16, minute = 56, second = 12, microsecond = 13, nanosecond = 16)
```

Convert something to Timestamp format (YYYY-MM-DD HH:MM:SS)
Microseconds and nanoseconds appear as decimals in the seconds field

```
d.year; d.month; d.day; d.minute; d.second; d.microsecond; d.nanosecond
```

Timestamp attributes

```
d.quarter; d.week; d.dayofweek; d.dayofyear; d.day_name(); d.month_name()
```

Timestamp attributes that do calculations

Period class

```
p = pd.Period("2017-01")
```

Basic Period creation

```
p.start_time
```

Period start time (with months: 1st day of month 00:00:00)

```
p.end_time
```

Period end time (with months: last day of month 23:59:59.999...)

Period class (cont)

```
p.asfreq("D")
```

Change period to days (called frequency) If a YYYY-MM period was given, new period is last day of month

```
q = p.asfreq("M")
```

Change period to months (called frequency) If a YYYY-MM-DD period was given, day is discarded and can't be retrieved.

```
pd.Period("2018-05").to_timestamp()
```

Period to timestamp

```
p = pd.Period("2017-05-23") + 2
```

Add two days to period

```
pd.Period("2017-05-23") - pd.Period("2017-05-28")
```

Returns time difference in days (chosen frequency)

Represents a period of time, not an instant. Months are the default period.

period_range class

```
p = pd.period_range(start = "2019-1-1T15:00", periods = 12, freq = "H")
```

Timedelta class

```
d = pd.Timedelta(weeks = 1, days = 2, hours = 3, minutes = 4, seconds = 5, milliseconds = 6, microseconds = 7, nanoseconds = 8)
```

```
pd.Timedelta("1 day 1 millisecond")
```

```
pd.Timedelta("2.3 hours")
```

Creation of Timedelta object

```
d.days; d.seconds; d.microseconds; d.nanoseconds
```

Timedelta attributes

```
pd.Timestamp(2019, 8, 25, 18, 49) + pd.Timedelta("3 hours")
```

Operations with timestamp

Used for making operations with Timestamps (subtraction, addition...)

date_range class

```
pd.date_range("August, 28 2018", periods = 5, freq = "M")
```

```
pd.date_range("August, 28 2018", periods = 5, freq = "D")
```

```
pd.date_range(start = date(2019, 8, 13), end = date(2019, 9, 21), freq = "W")
```

```
pd.date_range(start = "2018-01-1", end = "2019-12-31", freq = "MS")
```

from datetime import datetime

```
pd.date_range(start = datetime.today(), periods = 5, freq = "H")
```

```
pd.date_range(start = "2019-1-1", end = "2019-3-1", periods = 4)
```

```
pd.date_range(start = "2019-8-1", end = "2019-8-31", freq = "B")
```

```
pd.date_range(start = "2019-8-1", end = "2019-8-31", freq = "B") + pd.Timedelta("3 hours")
```

Índices formados por secuencias de fechas (DatetimeIndex). El rango podrá venir definido por una fecha de comienzo, una fecha de finalización, un número de períodos y una frecuencia (por tres de estos cuatro parámetros).

