Working with datetime in Pandas DataFrame

```
import pandas as pd
import numpy as np
from datetime import datetime
```

pandas.date_range() is one of the general functions in Pandas which is used to return a fixed frequency DatetimeIndex.

```
pd.date_range(start='2/2/2019', end='2/08/2019')
```

```
DatetimeIndex(['2019-02-02', '2019-02-03', '2019-02-04', '2019-02-05',
               '2019-02-06', '2019-02-07', '2019-02-08'],
              dtype='datetime64[ns]', freq='D')
```

```
#create a date range with timestamps of hourly frequency
date_rg=pd.date_range(start='1/1/2019', end='1/3/2019', freq='H')
date_rg
```

```
DatetimeIndex(['2019-01-01 00:00:00', '2019-01-01 01:00:00',
               '2019-01-01 02:00:00', '2019-01-01 03:00:00',
               '2019-01-01 04:00:00', '2019-01-01 05:00:00',
               '2019-01-01 06:00:00', '2019-01-01 07:00:00',
               '2019-01-01 08:00:00', '2019-01-01 09:00:00',
               '2019-01-01 10:00:00', '2019-01-01 11:00:00',
               '2019-01-01 12:00:00', '2019-01-01 13:00:00',
               '2019-01-01 14:00:00', '2019-01-01 15:00:00',
               '2019-01-01 16:00:00', '2019-01-01 17:00:00',
               '2019-01-01 18:00:00', '2019-01-01 19:00:00',
               '2019-01-01 20:00:00', '2019-01-01 21:00:00',
               '2019-01-01 22:00:00', '2019-01-01 23:00:00',
               '2019-01-02 00:00:00', '2019-01-02 01:00:00',
               '2019-01-02 02:00:00', '2019-01-02 03:00:00',
               '2019-01-02 04:00:00', '2019-01-02 05:00:00',
               '2019-01-02 06:00:00', '2019-01-02 07:00:00',
               '2019-01-02 08:00:00', '2019-01-02 09:00:00',
               '2019-01-02 10:00:00', '2019-01-02 11:00:00',
               '2019-01-02 12:00:00', '2019-01-02 13:00:00',
               '2019-01-02 14:00:00', '2019-01-02 15:00:00',
               '2019-01-02 16:00:00', '2019-01-02 17:00:00',
               '2019-01-02 18:00:00', '2019-01-02 19:00:00',
               '2019-01-02 20:00:00', '2019-01-02 21:00:00',
               '2019-01-02 22:00:00', '2019-01-02 23:00:00',
               '2019-01-03 00:00:00'],
              dtype='datetime64[ns]', freq='H')
```

```
pd.date_range(start='2/2/2019', periods=8)
```

```
DatetimeIndex(['2019-02-02', '2019-02-03', '2019-02-04', '2019-02-05',
               '2019-02-06', '2019-02-07', '2019-02-08', '2019-02-09'],
                      dtype='datetime64[ns]', freq='D')
```

Convert strings to datetime

```
d={'date': ['3/10/2000', '3/11/2000', '3/12/2000'],
                   'value': [2, 3, 4]}
df = pd.DataFrame(d)
print(df)
```

```
        date  value
0  3/10/2000      2
1  3/11/2000      3
2  3/12/2000      4
```

```
df['date'] = pd.to_datetime(df['date'])
df
```

	date	value
0	2000-03-10	2
1	2000-03-11	3
2	2000-03-12	4

Custom format

```
df = pd.DataFrame({'date': ['2016-6-10 20:30:0',
                            '2016-7-1 19:45:30',
                            '2013-10-12 4:5:1'],
                   'value': [2, 3, 4]})
df
```

	date	value
0	2016-6-10 20:30:0	2
1	2016-7-1 19:45:30	3
2	2013-10-12 4:5:1	4

```
df['date'] = pd.to_datetime(df['date'], format="%Y-%d-%m %H:%M:%S")
df
```

	date	value

Handle parsing error

```
df = pd.DataFrame({'date': ['3/10/2000', 'a/11/2000', '3/12/2000'],
                   'value': [2, 3, 4]})
df['date'] = pd.to_datetime(df['date'])
```

```
---------------------------------------------------------------------------
ParserError                               Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/pandas/_libs/tslib.pyx in pandas._libs.tslib.array_tc

                                14 frames
ParserError: Unknown string format: a/11/2000

During handling of the above exception, another exception occurred:

TypeError                                 Traceback (most recent call last)
TypeError: invalid string coercion to datetime for "a/11/2000" at position 1

During handling of the above exception, another exception occurred:

ParserError                               Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/dateutil/parser/_parser.py in parse(self, timestr, de
**kwargs)
    641
    642         if res is None:
--> 643             raise ParserError("Unknown string format: %s", timestr)
    644
    645         if len(res) == 0:

ParserError: Unknown string format: a/11/2000 present at position 1
```

SEARCH STACK OVERFLOW

```
df['date'] = pd.to_datetime(df['date'], errors='ignore')
df
```

	date	value
0	3/10/2000	2
1	a/11/2000	3
2	3/12/2000	4

```
df['date'] = pd.to_datetime(df['date'], errors='coerce')
df
```

	date	value
0	2000-03-10	2
1	NaT	3

Get year, month and day

```
df = pd.DataFrame({'name': ['Tom', 'Andy', 'Lucas','Pranav','Uma','Rahu','Kumar'],
                  'DoB': ['08-05-1997', '04-28-1996', '12-16-1995','12-18-1995','12-18-1996','11-16
df
```

	name	DoB
0	Tom	08-05-1997
1	Andy	04-28-1996
2	Lucas	12-16-1995
3	Pranav	12-18-1995
4	Uma	12-18-1996
5	Rahu	11-16-1995
6	Kumar	11-16-1999

```
df['DoB'] = pd.to_datetime(df['DoB'])
df
```

	name	DoB
0	Tom	1997-08-05
1	Andy	1996-04-28
2	Lucas	1995-12-16
3	Pranav	1995-12-18
4	Uma	1996-12-18
5	Rahu	1995-11-16
6	Kumar	1999-11-16

```
df['year']= df['DoB'].dt.year
df['month']= df['DoB'].dt.month
df['day']= df['DoB'].dt.day
df
```

	name	DoB	year	month	day
0	Tom	1997-08-05	1997	8	5
1	Andy	1996-04-28	1996	4	28
2	Lucas	1995-12-16	1995	12	16
3	Pranav	1995-12-18	1995	12	18
4	Uma	1996-12-18	1996	12	18
5	Rahu	1995-11-16	1995	11	16

Get the age from the date of birth

```
today = pd.to_datetime('today')
df['age'] = today.year - df['DoB'].dt.year

df
```

	name	DoB	year	month	day	age
0	Tom	1997-08-05	1997	8	5	26
1	Andy	1996-04-28	1996	4	28	27
2	Lucas	1995-12-16	1995	12	16	28
3	Pranav	1995-12-18	1995	12	18	28
4	Uma	1996-12-18	1996	12	18	27
5	Rahu	1995-11-16	1995	11	16	28
6	Kumar	1999-11-16	1999	11	16	24

```
df=df.set_index(['DoB'])
df
```

	name	year	month	day	age
DoB					
1997-08-05	Tom	1997	8	5	26
1996-04-28	Andy	1996	4	28	27
1995-12-16	Lucas	1995	12	16	28
1995-12-18	Pranav	1995	12	18	28
1996-12-18	Uma	1996	12	18	27
1995-11-16	Rahu	1995	11	16	28
1999-11-16	Kumar	1999	11	16	24

Improve performance by setting date column as the index

```
df
```

DoB	name	year	month	day	age
1997-08-05	Tom	1997	8	5	26
1996-04-28	Andy	1996	4	28	27
1995-12-16	Lucas	1995	12	16	28
1995-12-18	Pranav	1995	12	18	28
1996-12-18	Uma	1996	12	18	27
1995-11-16	Rahu	1995	11	16	28

7. Select data with a specific year and perform aggregation

```
df.loc['1996']
```

DoB	name	year	month	day	age
1996-04-28	Andy	1996	4	28	27
1996-12-18	Uma	1996	12	18	27

```
df.loc['1996','age'].sum()
```

```
54
```

```
df['1995'].groupby('month').sum()
```

```
<ipython-input-30-30d005d57776>:1: FutureWarning: Indexing a DataFrame with a datetimeli
  df['1995'].groupby('month').sum()
<ipython-input-30-30d005d57776>:1: FutureWarning: The default value of numeric_only in D
  df['1995'].groupby('month').sum()
```

month	year	day	age
11	1995	16	28
12	3990	34	56

Select data with a specific month or a specific day of the month

```
df.loc['1995-12']
```

DoB	name	year	month	day	age
1995-12-16	Lucas	1995	12	16	28
1995-12-18	Pranav	1995	12	18	28

```
cond = df.index.month==12
df[cond]
```

DoB	name	year	month	day	age
1995-12-16	Lucas	1995	12	16	28
1995-12-18	Pranav	1995	12	18	28
1996-12-18	Uma	1996	12	18	27

Select data between two dates

```
df.loc['1995' : '1997']
```

```
<ipython-input-58-f0fbd0cfb2f9>:1: FutureWarning: Value based partial slicing on non-monotoni
  df.loc['1995' : '1997']
```

DoB	name	year	month	day	age
1997-08-05	Tom	1997	8	5	26
1996-04-28	Andy	1996	4	28	27
1995-12-16	Lucas	1995	12	16	28
1995-12-18	Pranav	1995	12	18	28
1996-12-18	Uma	1996	12	18	27
1995-11-16	Rahu	1995	11	16	28